📖

Rappresentazione conoscenza

Date	@June 3, 2022 → June 7, 2022

Logica proposizionale

- è uno dei più semplici tipi di logica
- le formule non includono variabili

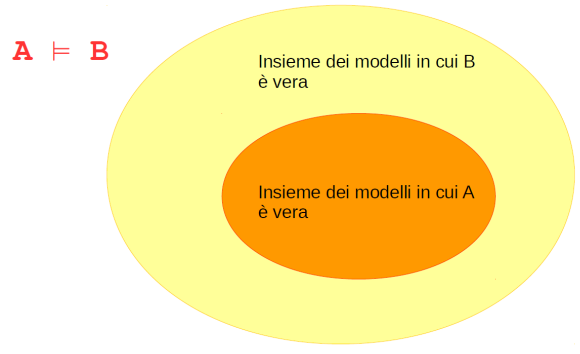
$$A \Rightarrow B \equiv \neg A \vee B$$

formula da ricordare, verrà usata in alcune dimostrazioni successive.

Qual è l’utilità?

> 📌 Rappresentare la conoscenza in modo tale che sia possibile applicarvi dei processi di ragionamento automatici (**inferenze**) per derivare informazioni, nuova conoscenza e per prendere decisione – in particolare per decidere quale azione eseguire.

$A \models B$ indica che da A consegue B, ad esempio: $(x + y = 4) \models (x + y < 5)$



Il fuoco è posto sui modelli in cui A è vera

L'**inferenza** è sintattica, lavora sulla struttura delle formule secondo il linguaggio di rappresentazione scelto, non si ferma sui modelli.

Modus ponens

💡 Da un'implicazione e dalla sua premessa, derivo la conseguenza

$$\frac{A \supset B \qquad A}{B}$$

$KB \models_i A$ indica che da A può essere inferita da KB

Come dimostrare una conseguenza logica?

▼ **Model Checking**

- enumero i possibili modelli
- seleziono quelli in cui KB è vera
- verifico che in tutti questi P sia vera

Procedimento molto **costoso**

▼ **Theorem proving**

permette di **usare regole di inferenza** per cercare una derivazione, senza costruire i modelli

Teorema di deduzione

- mette in relazione la conseguenza logica (semantica modelli delle formule) con l'implicazione (operatore della logica, formula della logica)

💡 Date due formule R e Q, $(R \models Q)$ se e solo se $(R \Rightarrow Q)$ è **valida** (ossia è una tautologia)

Dimostrazione per refutazione

💡 Date due formule R e Q, $(R \models Q)$ se e solo se $(R \wedge \neg Q)$ è **insoddisfacibile**

$$\frac{B \vee C \qquad \neg C \vee A}{A \vee B}$$

regola di risoluzione

$$\frac{C \qquad C \Rightarrow A}{A}$$

modus ponens

Algoritmo di traduzione in clausole

1) Eliminare la biimplicazione: $((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha))$
2) Eliminare l'implicazione: $(\neg\alpha \vee \beta)$
3) Portare il not all'interno (De Morgan ed eliminazione della doppia negazione):
   - $(\neg\alpha \vee \neg\beta)$ oppure $(\neg\alpha \wedge \neg\beta)$
   - α
1) Distribuire l'or sull'and dove possibile: $((\alpha \vee \beta) \wedge (\alpha \vee \gamma))$

Algoritmo di traduzione in CNF (conjuctive normal form)

Le clausole di Horn costituiscono le basi della **programmazione logica.**

Forward chaining

$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$

Fatti: A, B
Si vuole dimostrare Q

Questo è un grafo AND-OR.

Gli archi indicano gli AND e le frecce gli OR.

Da questo grafo si dedurrà che oltre ad A e B anche L ed M saranno veri: L perché AND di due proposizioni vere ed M perché anch'esso AND di due proposizioni vere.

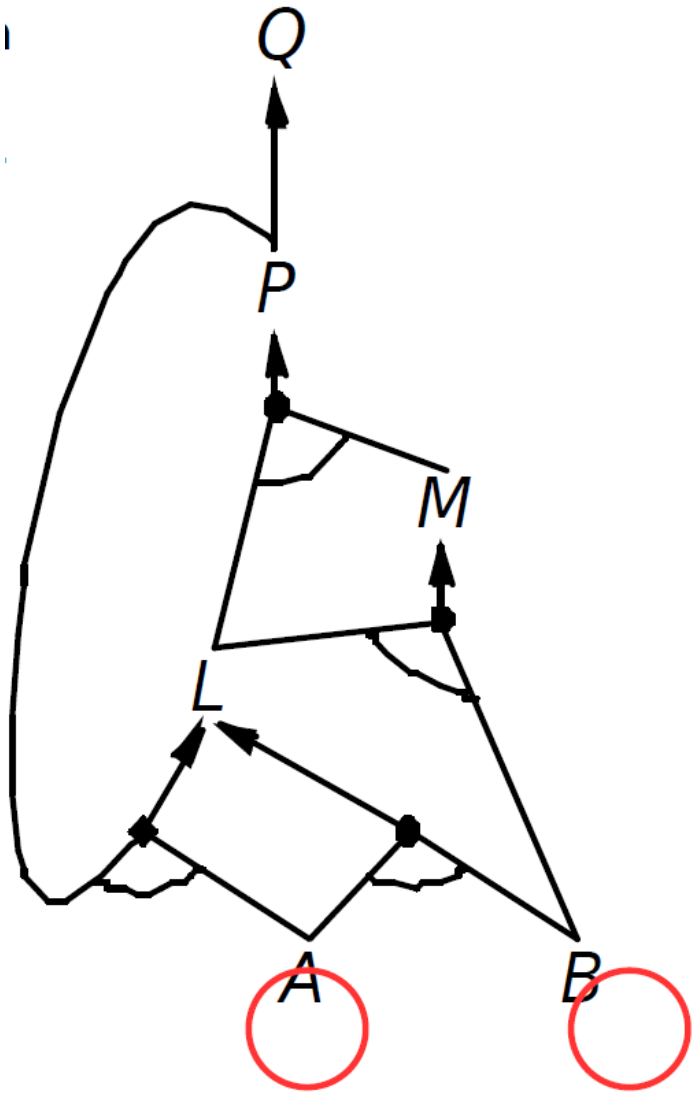


immagine che mostra il forward chaining avendo come fatti A e B (ossia A e B sono veri)

- un difetto del **forward chaining** è che è guidato dai dati e non usa l'informazione relativa al goal
- può attivare molte **inferenze inutili** ai fini della dimostrazione della formula in oggetto

Backward chaining

💡 Parte dalla formula da dimostrare (goal):
se risulta già vera termina restituendo true
altrimenti cerca clausole di Horn di cui **la formula è conclusione** e cerca di dimostrarne le premesse usando come informazione aggiuntiva i fatti noti

- realizza una forma di ragionamento guidato dagli obiettivi
- è **più efficiente** del forward chaining in quanto l'uso del goal focalizza la ricerca
- la complessità temporale è **meno che lineare**

Logica del prim’ordine

- **estensione della logica proposizionale**: si aggiunge la possibilità di esprimere relazioni fra oggetti
- contiene un **dominio**, cioè l’insieme degli oggetti del mondo considerati, e delle **relazioni** fra tali oggetti
- una **relazione** può essere un **predicato** (cattura una proprietà di un oggetto del dominio e vale T o F) o una **funzione** (oggetto del dominio)

Quantificatori

Al corso di sistemi intelligenti tutti sono intelligenti	$\forall x Partecipa(x, \text{SISINT}) \Rightarrow Intelligente(x)$
Al corso di sistemi intelligenti qualcuno è intelligente	$\exists x Partecipa(x, \text{SISINT}) \wedge Intelligente(x)$

- **Database semantics**: è la semantica usata nella programmazione logica e si basa su tre assunti
  - **Unicità dei nomi**:
    assumiamo che costanti diverse si riferiscano a oggetti del dominio diversi
  - **Closed-world assumption**:
    assumiamo che le formule atomiche delle quali non si conosce la verità siano false
  - **Domain closure**:
    un modello non contiene più elementi di quelli nominati dalle costanti

in FOL si utilizza la database semantics

Regole di istanziazione

Universale

- Data la formula **F :**
  ∀x (Partecipa(x, SISINT) ⇒ Intelligente(x))
- Date le sostituzioni alternative **θ1 = {x/Rufus}** e **θ2 = {x/Adele}** l’applicazione di UI permette di inferire rispettivamente:
  - **F θ1** è Partecipa(Rufus, SISINT) ⇒ Intelligente(Rufus)
  - **F θ2** è Partecipa(Adele, SISINT) ⇒ Intelligente(Adele)

UI

Esistenziale

- Data la formula:
  ∃x Corona(x) ∧ SullaTesta(x, John)
- Possiamo inferire **Corona(C1) ∧ SullaTesta(C1, John)** dove **C1** è un nome di costante nuovo inventato appositamente tramite un processo detto Skolemizzazione
- C1 non compare nella KB fino alla sua creazione
- **Attenzione**: usando la semantica standard di FOL, che non prevede unicità dei nomi, potremo poi inferire che **C1 = CoronaInglese**

EI

Differenze fra UI e EI

Utilizzando la UI: la nuova KB è **logicamente equivalente** a quella originaria

Utilizzando la EI, invece: la nuova KB **non è logicamente equivalente** a quella originaria ma **è soddisfacibile** se la prima lo era

Due processi per inferire deduzioni

Proposizionalizzazione

- applicare le regole di istanziazione per rimuovere i quantificatori
- applicare un algoritmo di inferenza per logica proposizionale
- è **inefficiente** perché *perde tempo* a creare istanze dell'implicazione che sono ininfluenti

Modus ponens generalizzato

Unificazione: algoritmo chiave di tutte le tecniche di inferenza sul prim'ordine

- richiede che **l'antecedente** dell'implicazione sia una **congiunzione**

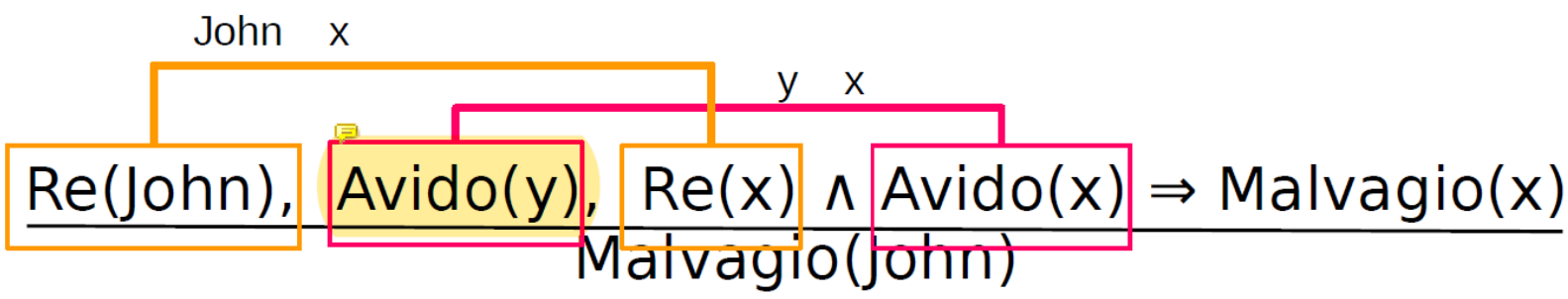
Modus Ponens Generalizzato (MPG)

$$\frac{p'_1,\ p'_2,\ \ldots,\ p'_n,\quad p_1 \wedge p_2 \wedge \ldots \wedge p_n \Rightarrow q}{q\theta}$$

Dove $p'_i\theta = p_i\theta$ per ogni $i \in [1,n]$

La regola ha come premesse n formule atomiche e una singola implicazione. La conclusione è il risultato dell'applicazione della sostituzione θ alla formula q, conseguenza dell'implicazione

Esempio



Questa conclusione si appoggia alla sostituzione {x/John, y/John}

Le clausole focalizzano la ricerca della sostituzione e permettono di ragionare direttamente in FOL

Nella proposizionalizzazione, di contro, si costruiscono sostituzioni usando in modo esaustivo l'intero vocabolario di costanti

---

Tradurre una KB FOL in CNF

FOL: ∀x [∀y Animale(y) ⇒ Ama(x, y)] ⇒ [∃y Ama(y, x)]

Tutti coloro che amano gli animali sono amati da qualcuno

1) **Elimina l'implicazione:**
   ∀x ¬ [∀y Animale(y) ⇒ Ama(x, y)] ∨ [∃y Ama(y, x)]
   ∀x ¬ [∀y ¬Animale(y) ∨ Ama(x, y)] ∨ [∃y Ama(y, x)]
2) **Sposta la negazione all'interno (¬ ∀ ≡ ∃¬):**
   ∀x [∃y Animale(y) ∧ ¬ Ama(x, y)] ∨ [∃y Ama(y, x)]
3) **Standardizzazione delle variabili in (∃y …) ∨ (∃y …):**
   ∀x [∃y Animale(y) ∧ ¬ Ama(x, y)] ∨ [∃z Ama(z, x)]
4) **Skolemizzazione (eliminazione degli esistenziali): …**

Skolemizzazione

📌 Sostituiamo ogni variabile quantificata esistenzialmente con una funzione che ha per argomenti tutte le variabili quantificate universalmente nel cui scope ricade.

Ingegneria della conoscenza

Tassonomie

📌 Standardizzare la rappresentazione di **categorie**, di introdurre **relazioni fra categorie** e di implementare meccanismi di eredità di proprietà fra categorie

$Member(X, Pallone) \Rightarrow Sferico(X)$ può essere un esempio di **proprietà**

Tramite le relazioni di sottoclasse le istanze di una classe **ereditano le proprietà** delle sovraclassi.

Ontologie

- è una forma più generale di conoscenza
- la tassonomia è una forma particolare di ontologia
- la tassonomia ha una forma ad **albero**, l'**ontologia** ha una forma a grafo

Si può suddividere in:

T-box

contiene una **concettualizzazione intensionale**, fatta di definizioni, specializzazioni, proprietà

Esempio:
Madre è sottoclasse di Donna,
$Madre(X) \Rightarrow Donna(X)$

A-box

riguarda istanze specifiche, è **estensionale**

Esempio:

Anna è una Madre:
$Madre(Anna)$

Alcuni tipologie di quesiti generici sono:

1) **Istanza appartiene a categoria?** Fido è un mammifero?
2) **Istanza gode di proprietà?** Fido può volare?
3) **Differenza fra categorie?** Quale differenza c'è fra rocce magmatiche e rocce sedimentarie?
4) **Identificazione di istanze?** Quali alberghi a tre stelle di Rimini offrono supporto tecnico ai ciclisti?

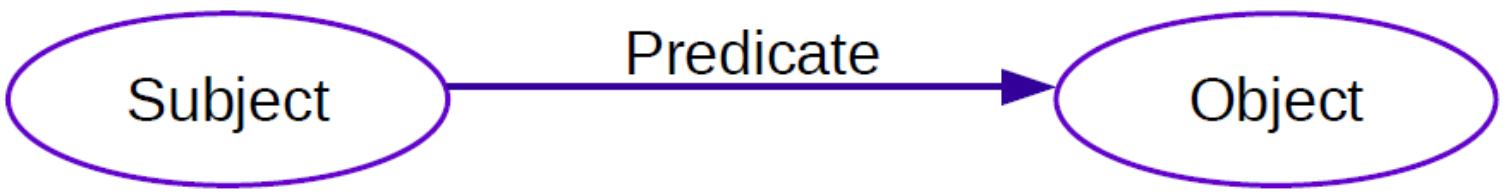
Diversi modi in cui si può interrogare una **rete semantica** (ontologia)

Semantic web

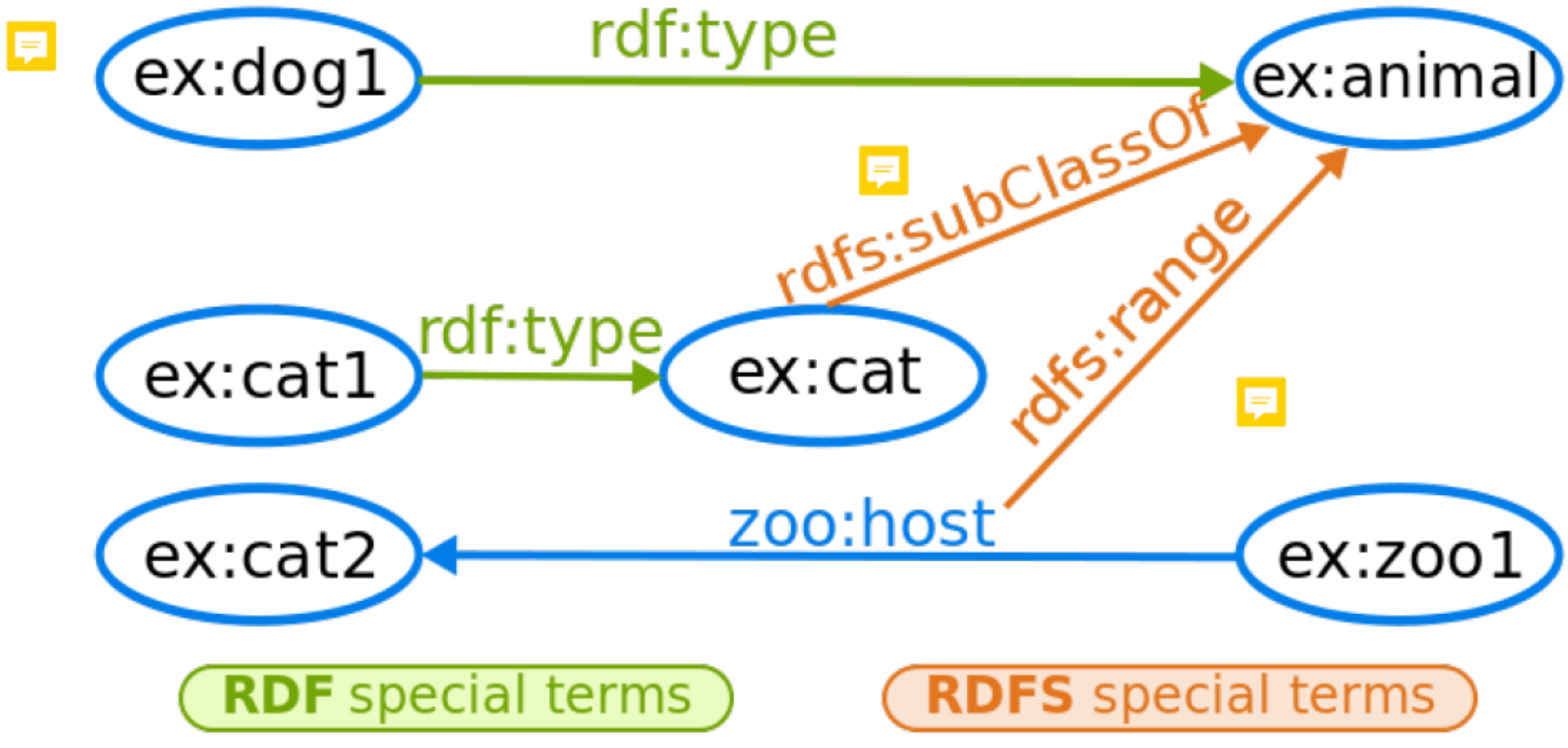
- estensione del WWW in cui il materiale pubblicato è arricchito da **metadati** che abilitano l'interpretazione, l'inferenza, l'interrogazione, l'elaborazione automatica

RDF sta per Resource Description Framework

- è un modello (e linguaggio) di rappresentazione



Esempio di IRI: International Resource Identifier



rdfs:subClassOf indica la relazione is-a

OWL sintassi

`ClassAssertion` corrisponde a `Member` , `SubClassOf` corriponde a `is-a`

Allineamento ontologico

- **Identical**: O1 e O2 sono la stessa ontologia (esempio: un'ontologia e la sua copia in un disco mirror)
- **Equivalent**: condividono vocabolario e assiomatizzazione ma sono espresse in linguaggi differenti (esempio: SKOS e RDF)
- **Extension**: O1 estende O2 quando tutti i simboli definiti in O2 sono preservati in O1 insieme alle loro proprietà e relazioni ma non vale il viceversa
- **Weakly-Translatable**: siano Osource e Odest due ontologie, è possibile tradurre espressioni Osource in espressioni Odest con perdita di informazione
- **Strongly-Translatable**: slide
- **Approx-Translatable**: slide

Relazioni fra ontologie

Ontologie vs database

- un fatto non contenuto in un DB è considerato falso (**closed world assumption**) in OWL 2 sarà mancante (**open world assumption**)
- OWL non richiede che le uniche proprietà di un individuo siano quelle della classe a cui appartiene
- Classi e proprietà possono avere definizioni multiple

Situation calculus

📌 Una rappresentazione logica che identifica i concetti base di:
+ **azione**: qualcosa che viene compiuto e influenza il mondo
+ **situazione**: stati derivanti dall'esecuzione di qualche azione
+ **fluente**: relazione o proprietà che può cambiare valore (fluire)
+ **predicato atemporale** (eterno): sono funzioni o predicati il cui calcolo non è influenzato dalle azioni

Fluenti

Esempio di **fluenti**:

- $Adjacent(R1, R2, s)$: $R1$ e $R2$ sono adiacenti nella situazione s
- $Holds(At(R, Loc), s)$: R si trova in posizione Loc nella situazione s

Azioni

Un'**azione è intesa come un oggetto intangibile**, prodotto da una funzione.

Azioni e situazioni

- $Do(Azioni, S)$: funzione che restituisce la **situazione raggiunta**, applicando la sequenza di azioni indicate a partire dallo stato indicato
- due situazioni sono **identiche** esclusivamente se sono originate dallo **stesso stato iniziale** applicando la stessa sequenza di azioni

Assioma di applicabilità

$$\forall params, s Applicable(Action(params), s) \Leftrightarrow Precond(params, s)$$

Esempio: $Applicable(go(X, Y), S) \Leftrightarrow At(X, S) \wedge Adjacent(X, Y)$, dove:

- $go(X, Y)$: **azione**, andare dalla posizione X alla posizione Y
- $At(Agente, X, S) \wedge Adjacent(X, Y)$ è la precondizione che ci dice che $go(X, Y)$ può essere eseguita a patto che l'agente sia in X (**fluente**) e che X sia adiacente a Y (**predicato atemporale**)

Assioma di effetto

$$\forall params, s Applicable(Azione(params), s) \Rightarrow Effects(params, Result(Action(params), s))$$

Esempio: $Applicable(go(X, Y), S) \Rightarrow At(Y, Result(go(X, Y), S))$, dove:

- l'effetto dell'azione applicabile $go(X, Y)$ è che nella **situazione risultante** dall'esecuzione di tale azione in s (identificata da $Result(go(X, Y), S)$) l'agente si trova alla posizione Y

Frame problem

📌 Dalla conoscenza di stato iniziale, assiomi di applicabilità e assiomi di effetto **non** è possibile derivare tutti i fatti che ci aspettiamo: non si riesce a rappresentare ciò che non viene modificato

Occorre introdurre dunque un modo per dire al sistema inferenziale che ciò che non è specificamente espresso come effetto è inteso **rimanere immutato.**

Ciò si fa tramite l'**assioma di stato successore.**

Definizioni

▼ Knowledge base (KB)

un insieme di formule espresse in un linguaggio per la rappresentazione della conoscenza possedute dall'agente. Può cambiare nel tempo.

▼ Background knowledge

conoscenza iniziale di un agente

▼ `tell` ed `assert`

`tell` aggiunge nuove formule, `assert` effettua interrogazioni

▼ Informazione

ciò che un dato rappresenta

▼ Linguaggio di rappresentazione

è lo strumento che consente di rappresentare la conoscenza in una forma su cui è possibile applicare forme di ragionamento automatico (**inferenza**).

▼ Modello

un modello fissa i valori di verità delle formule

▼ Conseguenza logica

è una relazione fra due formule che dice che in tutti i modelli in cui la prima formula è vera, è vera anche la seconda

▼ Validità

Una formula P è valida se è **vera in tutti i modelli**

▼ Insoddisfacibilità

Una formula P è insoddisfacibile se è **falsa in tutti i modelli**

▼ Soddisfacibilità

Una formula P è soddisfacibile se **esiste qualche modello in cui è vera**

▼ Inferenza

è il processo con il quale da una proposizione, accolta come vera, si passa a una seconda proposizione la cui verità è derivata dalla prima

▼ Semantica della logica

la **semantica** definisce le regole con cui si calcolano i valori di verità di tutte le formule

▼ Logica monotona

nelle logiche monotone l'aggiunta di informazione non invalida mai le conclusioni precedenti

▼ Clausola di Horn

una clausola di Horn è una **disgiunzione di letterali** di cui al più uno è positivo

▼ Forward chaining

permette di derivare una query data da un singolo simbolo proposizionale da una KB costituita da clausole di Horn

▼ Interpretazione in FOL

l'interpretazione è il fondamento per determinare il valore di verità delle formule. È un'associazione fra i simboli e gli oggetti del dominio del discorso

▼ Termine ground

un termine è ground quando non contiene variabili

▼ Lifting

trasportare le regole di inferenza usate nei linguaggi più poveri (come quello proposizionale) in quelli più complessi

▼ Teorema di Herbrand

📌 se una formula è **conseguenza logica** della base di conoscenza originaria (del prim'ordine) allora partendo dalla base di conoscenza proposizionalizzata esiste una dimostrazione finita della sua verità

▼ Semidecidibile

non esiste un algoritmo per dimostrare che una certa conseguenza non vale

▼ Clausole di Horn FOL

disgiunzioni di letterali di cui al più uno è positivo **oppure** implicazioni il cui antecedente è un congiunzione di letterali

▼ Lifting della fattorizzazione

due letterali sono ridotti ad uno non se sono uguali ma **se sono unificabili**. L'unificatore va applicato alle clausole intere

▼ Member

Member(P, PalloneCalcio) è un **predicato** che restituisce vero se P è un elemento della categoria PalloneCalcio, P rappresenta un'**istanza** della categoria PalloneCalcio

▼ Is-a

Is-a è un **predicato** che esprime una relazione di sottocategoria fra categorie

▼ Tassonomia

Una tassonomia è l'organizzazione delle categorie risultante da un insieme di regole di sottoclasse

▼ Categorie disgiunte

quando non hanno istanze comuni

▼ Categorie esaustive

quando non esiste nessuna istanza di c che non sia istanza di $S1$ o $S2$.
dove $S1$ ed $S2$ sono due categorie di una sovraclasse

▼ Partizione

Un insieme di categorie costituiscono una **partizione** quando sono esaustive e disgiunte.

▼ Part-of

alcune categorie sono parte di altre, esempio: Part-of(Leg, Table).
la relazione part-of gode della **proprietà transitiva**

▼ Allineamento ontologico

combinare concettualizzazioni sviluppate separatamente e indipendentemente.

▼ Assioma stato successore

Azione applicabile $\Rightarrow$ (fluente vero nella situazione risultante $\Leftrightarrow$(l'azione lo rende vero $\vee$ era vero e l'azione non l'ha reso falso))