$\leftrightarrow \leftarrow \rightarrow \neg \vee \wedge \subseteq \neq \cup \in \notin$

Lezione 5

LP e DataLog

- LP (Logica Proposizionale)
  - Abbiamo dato sintassi e semantica delle ***formule*** di LP
  - Oggi vediamo
    - KB: rappresentazione di conoscenze in LP
    - SR: sistema di ragionamento per LP

5.1. LP: teorie (KB) e modelli

- **Una teoria** KB è un insieme di formule
  - le teorie fanno parte della sintassi: sono la sintassi in cui esprimere una base di conoscenza KB
  - le formula di KB sono dette assiomi della teoria
- Un **modello** di KB è un'interpretazione I tale che
  I |= F per ogni F appartenente a KB
  - I modelli fanno parte della semantica: corrispondono alle realtà rappresentate dalla teoria

Notazione: I(A) = v *si scrive* I |= A

Esercizio

- KB1 = {nuvolo, ventoso, in_barca}
- KB2 = { nuvolo, ventoso,
  ventoso $\wedge$ $\neg$temporale $\rightarrow$ in_barca }
- I = {nuvolo,ventoso,in_barca} è modello di KB1 e di KB2?
- Dare una I che non sia modello di KB1 ma sia modello di KB2.
- KB1 ha I come (unico) ***modello minimo***. (EC)
  A quale principio non logico visto nelle prime lezioni corrisponde il considerare solo il modello minimo?
- Se l'agente deve simulare un patito della barca a vela, quale delle due KB è più "intelligente"?

5.1.1. Conseguenze logiche e dimostrazioni

- Una formula F è **conseguenza logica** di KB:
  KB |= F sse, per ogni modello I di KB, I |= F
  - la nozione di conseguenza logica fa parte della semantica; riguarda le proprietà vere in tutte le situazioni modellate da KB
- Un calcolo è un insieme di regole di dimostrazione. Una ***dimostrazione*** è una successione di passaggi certificabili in base alle ***regole del calcolo***, che porta dagli ***assiomi*** di KB ad un ***teorema*** F:
  KB |- F (da KB si dimostra F) sse esiste una dimostrazione del teorema F che usa gli assiomi di KB;

5.1.2. Calcoli = sistemi di ragionamento per LP

TEOREMI (ripasso da logica)

- **Validità:** KB |- F ➔ KB |= F
- **Completezza:** KB |= F ➔ KB |- F
- Per i teoremi di validità e completezza, un ***calcolo*** per la logica proposizionale classica è un **SR** (Sistema di Ragionamento) per LP:
  - **Domande:** F ? (cioè KB |= F?)
  - **Risposte:** SI / NO
- FATTO: in LP KB |= F è decidibile con complessità esponenziale (se KB è finita).

Dove siamo: concetti, notazioni e teoremi da conoscere

- ***Interpretazione*** I : ***P*** $\rightarrow$ {**v**,**f**} o I $\subseteq$ ***P***
- ***Verità*** di una formula F in I: I |= F
- Interpretazione I ***modello*** di una teoria KB : I |= KB
- Modelli (e interpretazioni) di Herbrandt
- Formula F ***conseguenza logica*** di una teoria KB : KB |= F
- ***Dimostrazione:*** successione di passaggi certificabili in base alle ***regole di un calcolo***, che porta dagli ***assiomi*** di KB ad un ***teorema*** F:
  KB |- F sse esiste una dimostrazione di F
- ***Validità:*** KB |- F ==> KB |= F
- ***Completezza:*** KB |= F ==> KB |- F
- ***Domande:*** F?
- ***Risposte:*** SI/NO

5.2. DataLog Proposizionale (DLP)

- Sottosistema di LP, primo passo verso DataLog
- Il calcolo per DLP verrà esteso a DataLog e Prolog
  - Sinonimi di “calcolo”:
    - Sistema di ragionamento (SR)
    - “motore inferenziale”
    - …

La sintassi di DLP

- *Atomi:* atomiche (simboli proposizionali ***P***)
- *Corpo* (Body): congiunzione di *atomi* B1 $\wedge$ .. $\wedge$ Bn
  Es. sereno $\wedge$ ventoso
- *Clausola Definita:*
  - *Fatto:* *Atomo*
    Es: soleggiato.
  - *Regola:* *Atomo* $\leftarrow$ *Corpo*
    Es. in_mare $\leftarrow$ sereno $\wedge$ ventoso.
    Atomo: TESTA (Head) della clausola.

- *Base Conoscenza* (o *Programma*):
  insieme di *Clausole Definite*
  ES:
  soleggiato $\leftarrow$ sereno.
  soleggiato $\leftarrow$ nubi_passeggere.
  fresco $\leftarrow$ nubi_passeggere.
  fresco $\leftarrow$ ventoso.
  benessere $\leftarrow$ fresco $\wedge$ soleggiato.
  ventoso.
  nubi_passeggere.

5.2.1. SR: sistema di ragionamento

Regola di ragionamento basata su Modus Ponens

$$\frac{B1, \ldots , Bn \qquad A \leftarrow B1\wedge .. \wedge Bn}{A}$$

Validità della regola: in ogni interpretazione I, se le premesse (sopra la riga) sono vere in I, allora la conseguenza è vera in I

IL CALCOLO: LE SUE PROVE

KB = 1. Clausola_1. ….. n. Clausola_n.

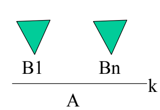
Base

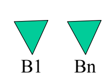
$$\frac{}{A}\,i$$

i. A fatto di KB

Passo

k. A $\leftarrow$ B1$\wedge$.. $\wedge$Bn
regola di KB





ricorsivamente, prove

Esempio di prova

1. soleggiato ← sereno.
2. soleggiato ← nubi_passeggere.
3. fresco ← nubi_passeggere.
4. fresco ← ventoso
5. benessere ← fresco ∧ soleggiato.
6. nubi_passeggere.

$$\dfrac{\dfrac{\dfrac{}{\text{nubi\_passeggere}}\,6}{\text{fresco}}\,3 \qquad \dfrac{\dfrac{}{\text{nubi\_passeggere}}\,6}{\text{soleggiato}}\,2}{\text{benessere}}\,5$$

Validità e completezza

- DEF. KB |- A sse esiste un albero di prova con radice A.
- Validità: KB |- A ➔ KB |= A
  - Dim. Per induzione sulla profondità della prova (lavagna)
- Completezza: KB |= A ➔ KB |- A
  - Dim. Vedremo.

5.2.2. Motore inferenziale. Procedura non deterministica Top Down (o Goal-Oriented o backward)

Ingresso: **KB** + ***Goal: atomo A da dimostrare***.

- Albero Parziale AP := A?
- Fino a che AP contiene un F? e nessun F?FAIL:
  - ***scegli*** un F?
  - Se esiste una clausola con testa F

    prendi k. F ← B1∧.. ∧Bn

    e applica la regola k all'indietro (vedi esempio)

    altrimenti

    marca F? con FAIL (cioè F?FAIL)

Non determinismo

- ***Scegli:*** don't-care
  - Fair selection rule (evitare starvation)
- ***Prendi***
  - Non determinismo don't-know (un oracolo può fare le scelte giuste)
  - Procedendo deterministicamente: necessità di BACKTRAKING (alberi di ricerca)
- THEO. La ricerca di una prova ha "non-determinisicamente successo" se e solo se la prova esiste

1. soleggiato ← sereno.
2. soleggiato ← nubi_passeggere.
3. fresco ← nubi_passeggere.
4. fresco ← ventoso
5. benessere ← fresco ∧ soleggiato.
6. nubi_passeggere.

Prendo 5

benessere?

1. soleggiato ← sereno.
2. soleggiato ← nubi_passeggere.
3. fresco ← nubi_passeggere.
4. fresco ← ventoso
5. benessere ← fresco ∧ soleggiato.
6. nubi_passeggere.

Scelgo soleggiato?
Prendo 2.
(e se prendessi 1?)

$$\dfrac{\textbf{fresco?} \qquad \textbf{soleggiato?}}{\text{benessere}}\,5$$

1. soleggiato ← sereno.
2. soleggiato ← nubi_passeggere.
3. fresco ← nubi_passeggere.
4. fresco ← ventoso
5. benessere ← fresco ∧ soleggiato.
6. nubi_passeggere.

Scelgo fresco?
Prendo 3
(se prendessi 4?)

$$\dfrac{\textbf{fresco?} \qquad \dfrac{\textbf{nubi\_passeggere?}}{\text{soleggiato}}\,2}{\text{benessere}}\,5$$

1. soleggiato ← sereno.
2. soleggiato ← nubi_passeggere.
3. fresco ← nubi_passeggere.
4. fresco ← ventoso
5. benessere ← fresco ∧ soleggiato.
6. nubi_passeggere.

Andate avanti voi, si
Arriva a

$$\dfrac{\dfrac{\textbf{nubi\_passeggere?}}{\text{fresco}}\,3 \qquad \dfrac{\textbf{nubi\_passeggere?}}{\text{soleggiato}}\,2}{\text{benessere}}\,5$$

1. soleggiato ← sereno.
2. soleggiato ← nubi_passeggere.
3. fresco ← nubi_passeggere.
4. fresco ← ventoso
5. benessere ← fresco ∧ soleggiato.
6. nubi_passeggere.

$$\dfrac{\dfrac{\dfrac{}{\text{nubi\_passeggere}}\,6}{\text{fresco}}\,3 \qquad \dfrac{\dfrac{}{\text{nubi\_passeggere}}\,6}{\text{soleggiato}}\,2}{\text{benessere}}\,5$$

5.2.3. Motore inferenziale: procedura Bottom-Up e modelli minimi

La procedura Bottom-Up genera dal basso tutti i fatti deducibili per ***Modus Ponens***:

$$\dfrac{A \leftarrow B1 \wedge .. \wedge Bn \qquad B1, \ldots, Bn}{A}$$

A ***conseguenza immediata*** di B1, .., Bn
e della la regola A ← B1∧.. ∧Bn

Conseguenza immediata e procedura bottom-up

- Operatore di conseguenza immediata
  Data KB e un'interpretazione I ⊆ ***P***:
  - T_{KB} (I) = I ∪ {H | H è un fatto di KB
    o una conseguenza immediata di I e di
    almeno una regola di KB)}

PROCEDURA

I := {};

while T_{KB}(I) ≠I do I := T_{KB} (I).

Esempio.

soleggiato ← sereno.
soleggiato ← nubi_passeggere.
fresco ← nubi_passeggere.
fresco ← ventoso.
benessere ← fresco ∧ soleggiato.
ventoso.
nubi_passeggere

TKB({}) = I_0 = {ventoso, nubi_passeggere}

TKB (I_0) = I_1 = {ventoso, nubi_passeggere, soleggiato, fresco}

TKB (I_1) = I_2 = {ventoso, nubi_passeggere, soleggiato, fresco, benessere}

TKB (I_2) = I_2 STOP

Dimostrabilità bottom-up: proprietà

Il risultato della procedura si indica con KB* :

KB* = I finale

Theo. Validità: A ∈ KB* ➔ KB |= A

Theo. Completezza: KB |= A ➔ A ∈ KB*

Theo. Esiste una prova di A, cioè KB |- A, sse A ∈ KB*

- ***Complessità:*** lineare nella dimensione di KB
  - ogni clausola è usata al più una volta per aggiungere un atomo
- ***Minimo punto fisso e minimo modello:***

  Teorema. KB* è un punto fisso di T: T(KB*) = KB*

  Teorema. KB* è il minimo punto fisso

  Teorema. KB* è il modello minimo di KB.

- NOTA. La completezza segue dal fatto che KB* è il modello minimo. Sapreste fare la dim.? (indicazioni a lezione)
- NOTA: ricordate l'assunzione del mondo chiuso? L'assunzione del modello minimo ha a che fare? (ec)

Dove siamo

PRIMO RRS

- DLP (DataLog Proposizionale)
  - Le prove
  - Algoritmo top-down (backward, goal oriented)
    - Non determinismo don't care e don't know
  - Algoritmo bottom-up (forward)
    - KB* come minimo punto fisso e modello minimo
    - Sul mondo chiuso
  - Validità e completezza dimostrate

5.3. Estensione a DATALOG

- Tutte le proprietà viste e quanto detto sulle procedure bottom up e top down si estende pari pari a DATALOG

5.3.1. DataLog: assunzioni

- **IR** (Individui e Relazioni): la conoscenza di un agente è descrivibile in termini di individui e relazioni
- **DK** (Definite Knowledge): la conoscenza di un agente è fornita da proposizioni definite e positive
  - Definite: non vaghe, escludo la disgiunzione
  - positive: escludo la negazione
- **SE** (Static Environment): l'ambiente non cambia (non c'è il tempo)
- **FD** (Finite Domain): l'universo di cui si parla è costituito da un insieme finito di individui.

Esercizio

- Ambito di problema:
  Un robot si muove in un ambiente formato da stanze in comunicazione fra loro e, tutte le sere, deve rimettere al loro posto gli oggetti che sono stati spostati durante il giorno. Conta gli spostamenti effettuati e li segna su file, che serve da archivio storico.
- Date le assunzioni di DataLog, individuare cosa si può modellare.
- Dare almeno un aspetto non modellabile usando DataLog, indicando quale delle assunzioni lo impedisce.

5.3.2 DataLog: la sintassi

- *Costante:* formata da lettere o cifre, inizia con una lettera minuscola o è numerica; es: pippo33, il_cane
- *Variabile:* formata da lettere, cifre, _, inizia con maiuscola o _ es: X12, _pippo
- *Simbolo di Predicato:* stessa sintassi delle costanti non numeriche
- *Termine:* *costante* o *variabile*
- *Atomo:* p(t1,..,tn), con p *predicato* e t1,..,tn *termini*
  - esempio: padre(mario,X)
- *Body, Clausola definita, testa, KB :*
  - come nel proposizionale

Esempio

- **In un cortile** ci sono il **gatto felix**, il **cane fido** e un albero. Il gatto, quando **vede** il cane, **scappa sull' albero**.
- I colori suggeriscono l'astrazione:
  - Costanti: scritte ***in rosso***
  - relazioni (in blu)
    - **vede** binaria
    - **in_cortile, su_albero, scappa, gatto, cane** unarie
- **Esempi di clausole**:

  in_cortile(gatto). (ec)

  scappa(X) $\leftarrow$ gatto(X) $\wedge$ vede(X,Y) $\wedge$ cane(Y).

Esercizio: i seguenti colori suggeriscono un'astrazione diversa:

In un **cortile** ci sono il **gatto felix**, il **cane fido** e un **albero**. Il gatto, quando **vede** il cane, **scappa sull' albero**.

Dare l'alfabeto datalog corrisopondente all'astrazione suggerita.

5.3.3. DataLog: la semantica Interpretazioni

- Interpretazione I = (D, Φ, π)

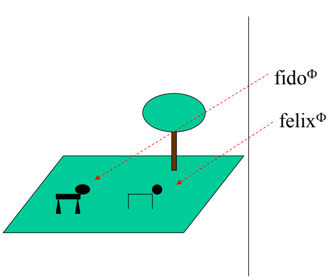
  D dominio di individui

  Φ : simbolo di costante c $\mapsto$ elemento $c^{\Phi} \in$ D

  π : simbolo di predicato n-ario p $\mapsto$ (funzione p^{π} : $D^n \rightarrow \{\mathbf{v},\mathbf{f}\}$)

Esempio

In un cortile ci sono il **gatto felix**, il **cane fido** e un albero.
Il gatto, quando **vede** il cane, **scappa sull' albero**.



$\text{gatto}^{\pi}(\text{felix}^{\Phi})$ = **V**

$\text{in\_cortile}^{\pi}(\text{fido}^{\Phi})$ = **V**

$\text{su\_albero}^{\pi}(\text{fido}^{\Phi})$ = **F**

ecc

5.3.3.1. Interpretazioni (modelli) di Herbrandt

- Un'interpretazione di Herbrandt è un insieme di ***atomi ground***
  - Un atomo è ground (o ***chiuso***) se non contiene variabili
  - Un atomo con variabili è detto ***aperto***
- Per poter rappresentare le interpretazioni alla Herbrandt in Datalog sono necessarie le ipotesi di (ec)
  - ***nome unico*** costanti diverse rappresentano individui diversi
  - ***mondo chiuso per gli individui:*** ogni individuo del dominio D è rappresentato da una costante
- *Sotto queste ipotesi ha senso confondere, nella semantica interna, gli individui con le costanti che li denotano; è quanto avviene nelle interpretazioni di Herbrandt*

- **DEF.** ***Interpretazione di Herbrandt per KB:***

$H \subseteq$ Atomi_Ground(KB)

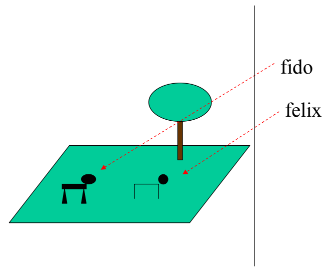
- H rappresenta l'interpretazione $I_H = (D, \Phi, \pi)$:
  - D = insieme delle costanti
  - Per ogni costante c, $c^{\Phi} = c$
  - Per ogni p n-ario, $p^{\pi}(c1,..,cn)=$**V** sse $p(c1,..,cn) \in H$

- NOTA. I modelli di Herbrandt forniscono di una KB forniscono la semantica "interna", da distinguersi da quella "esterna"

ESEMPIO

SEMANTICA ESTERNA

SEMANTICA INTERNA,
Come modello di Herbrandt



cane(fido)
gatto(felix)
in_cortile(fido)
in_cortile(felix)
vede(fido,felix)

5.3.3.2. Datalog: la semantica delle formule

- Diamo la semantica in versione semplificata, assumendo come interpretazioni quelle di Herbrandt
- Si veda il testo per la versione più generale, data usando la nozione di assegnamento di valori alle variabili
- Quella più generale si usa quando vi siano individui non denotati da costanti.

A) Semantica per le clausole ground

- Si usa la semantica già vista per il caso proposizionale, usando come simboli proposizionali gli atomi ground;
  - un atomo ground è vero in un'interpretazione di Herbrandt H sse appartiene ad H
  - i connettivi $\leftarrow$ $\wedge$
    si interpretano con le solite tavole di verità
- Per indicare che una clausola ground C è vera in un'interpretazione H scriveremo,

  $H \models C$

H = { cane(fido), gatto(felix), in_cortile(fido), in_cortile(felix), vede(fido,felix) } (ec)

$H \models$ in_cortile(fido) $\leftarrow$ vede(felix,fido) ?

$H \models$ vede(felix,fido) $\leftarrow$ vede(fido,felix) ?

B. Semantica per le clausole aperte

- Prendiamo la clausola:
  scappa(X) $\leftarrow$ gatto(X) $\wedge$ vede(X,Y) $\wedge$ cane(Y)
- È sottinteso che le variabili X,Y siano quantificate universalmente: per ogni X,Y …
- Se il dominio D è {felix, fido}, la clausola è vera sse sono vere le sue istanze ground
  scappa(felix) $\leftarrow$ gatto(felix) $\wedge$ vede(felix,felix) $\wedge$ cane(felix)
  scappa(felix) $\leftarrow$ gatto(felix) $\wedge$ vede(felix,fido) $\wedge$ cane(fido)
  scappa(fido) $\leftarrow$ gatto(fido) $\wedge$ vede(fido,felix) $\wedge$ cane(felix)
  scappa(fido) $\leftarrow$ gatto(fido) $\wedge$ vede(fido,fido) $\wedge$ cane(fido)

LA DEFINIZIONE FORMALE

- Istanza ground: si ottiene sostituendo uniformemente tutte le variabili con costanti

- **DEF.** Sia una clausola aperta;

  $H \models C(X_1,..,X_n)$

  sse, per ogni istanza ground $C(c_1,..,c_n)$, $H \models C(c_1,..,c_n)$

5.3.3.2. DataLog: i modelli di Herbrandt di una KB

- Una Base di Conoscenza è un insieme KB di clausole aperte, in un linguaggio con un dato insieme di costanti e simboli di predicato
- **DEF.** I modelli di Herbrandt di KB sono tutte e sole le interpretazioni di Herbrandt H tali che

  $H \models KB$

Esercizio

scappa(X) $\leftarrow$ gatto(X) $\wedge$ vede(X,Y) $\wedge$ cane(Y).
gatto(felix).
cane(fido).

- {gatto(felix)} è modello?
- {cane(fido), gatto(felix)} è modello?
- {cane(fido), gatto(felix), vede(felix,fido)} è modello?
- {cane(fido), gatto(felix), vede(felix,fido), scappa(felix)} è modello?
- {cane(fido), gatto(felix), vede(fido,felix)} è modello?
- Qual è il modello minimo?