Intelligenza Artificiale I

Logica formale

Introduzione alla logica del primo ordine

Marco Piastra

Limiti del linguaggio proposizionale

- *Esempio*:

  "Ogni essere umano è mortale"
  "Socrate è un essere umano"

  "Socrate è mortale"

  Il legame logico (intuitivo) è evidente

(Essere umano) implica (essere mortale)	$A \rightarrow B$	Schema del	$\varphi \rightarrow \psi$
Socrate è un essere umano	C	*modus ponens*	φ
Socrate è mortale	D		ψ

  Nella traduzione logico-proposizionale A, B, C e D non hanno alcun legame

- Altri esempi 'intraducibili':

  "Quando capra e cavolo stanno sulla stessa riva, la capra mangia il cavolo"

  "Il cacciatore sente una brezza quando si trova sul ciglio di una trappola"

Logica del primo ordine

- **Logica proposizionale**

  Rappresentazione del mondo come sulla base di **fatti** atomici (proposizioni)

- **Logica del primo ordine**

  Una base più complessa e articolata per la rappresentazione

  Oggetti

  persone, numeri, costruzioni, colori, storie, percorsi, pezzi di legno, …

  Proprietà (di oggetti) e **relazioni** (tra oggetti)

  rosso, grande, primo, fratello di, maggiore di, compreso tra, …

  Funzioni (di oggetti)

  successore, padre di, …

Linguaggio del primo ordine

- Simboli del linguaggio L_{PO}

  Costanti individuali (singoli oggetti)

    Esempi: *Socrate*, *Sfera1*, *MickeyMouse*, *Amelia*, *bianco*, *0* ...

    Genericamente indicati con: a, b, c, …

  Predicati

    Un predicato ha un nome ed numero di argomenti prestabilito (arità, *arity*)

    Esempi: *Red*/1, *Large*/1, *GreaterThan*/2, =/2

      Si identificano in base ad entrambi: *Large*/1 e *Large*/2 sono predicati <u>diversi</u>

    Genericamente indicati con: $P/1$, $Q/1$, $R/2$, …

Convenzionalmente, per il predicato binario = si usa la forma infissa (p.es. '$a = b$')

  Funzioni

    Una funzione ha un nome ed numero di argomenti prestabilito (arità, *arity*)

    Esempi: *sqrt*/1, *colorOf*/1, *greatestCommonDivisor*/2 ...

      Si identificano in base ad entrambi: *colorOf*/1 e *colorOf*/2 sono predicati <u>diversi</u>

    Genericamente indicate con: $f/1$, $g/1$, $h/2$, …

Linguaggio

Linguaggio del primo ordine (2)

- Ulteriori simboli del linguaggio L_{PO}

  Variabili

  Genericamente indicati con: $x, y, z, \ldots$

  Connettivi

  Gli stessi della logica proposizionale: $\neg, \rightarrow, \wedge, \vee, \leftrightarrow$

  Parentesi e virgola

  ( ) ,

  Quantificatori

  Universale: $\forall$

  Esistenziale: $\exists$

  Un quantificatore, una (sola) variabile
  Esempi: $\forall x$, $\exists y$, $\forall x \forall y$

Si dice linguaggio **del primo ordine** in quanto
i quantificatori agiscono solo sulle variabili x, y, z ...,
e non su **predicati** o **funzioni**
(*In una logica del secondo ordine si hanno formule del tipo*: $\exists F\ F(a,b)$)

Traduzione in formule di L_{PO}

- "Essere fratelli significa essere parenti"

  $\forall x \forall y\ (Fratello(x, y) \rightarrow Parente(x, y))$

- "La relazione di parentela è simmetrica"

  $\forall x \forall y\ (Parente(x ,y) \leftrightarrow Parente(y, x))$

- "Una madre è un genitore di sesso femminile"

  $\forall x \forall y\ (Madre(x, y) \leftrightarrow (Genitore(x, y) \land Femmina(x)))$

- "Un cugino è figlio di un fratello o una sorella di uno dei genitori"

  $$\forall x \forall y\ (Cugino(x, y) \leftrightarrow \\ \exists z \exists w\ (Genitore(z, x) \land Genitore(w, y) \land (Fratello(z, w) \lor Sorella(z, w))))$$

- "Ciascuno ha una madre"

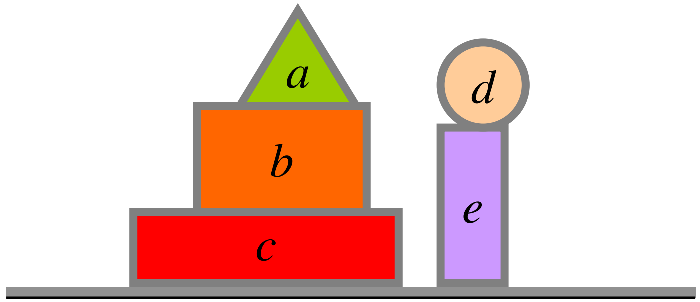
  $\forall x \exists y\ Madre(y, x)$

  Occorre fare attenzione all'ordine dei quantificatori:

  $\exists y \forall x\ Madre(y, x)$

  "Esiste una madre di tutti"

Descrivere un mondo



- Insiemi

  $Pyramid(a)$
  $Parallelepiped(b) \land Parallelepiped(c) \land Parallelepiped(e)$
  $Sphere(d)$
  $Ontable(c) \land Ontable(e)$
  $Clear(a) \land Clear(d)$

- Funzioni

  $(colorOf(a) = green) \land (colorOf(b) = orange) \land (colorOf(c) = red)$

- Relazioni

  $Above(a,b) \land Above(b,c) \land Above(a,c) \land Above(d,e)$

- Regole (definizioni)

  $\forall x \forall y\ (On(x,y) \leftrightarrow (Above(x,y) \land \neg \exists z\ (Above(x,z) \land Above(z,y)))$
  $\forall x\ (Ontable(x) \leftrightarrow \neg \exists z\ Above(x,z))$
  $\forall x\ (Clear(x) \leftrightarrow \neg \exists z\ Above(z,x))$

Termini e formule atomiche

- **Termini**

  Ogni *variabile* o *costante individuale* è un **termine**

  Se f è un *simbolo funzionale* a *n* argomenti e $t_1, \ldots, t_n$ sono **termini**, allora $f(t_1, \ldots, t_n)$ è un **termine**

  Un termine **chiuso** (*ground*) non contiene variabili

- **Formula atomica**

  Se P è un *simbolo predicativo* a *n* argomenti e $t_1, \ldots, t_n$ sono **termini**, allora $P(t_1, \ldots, t_n)$ è una **formula atomica**

Esempi:

$$\underbrace{\underbrace{Sorella}_{\text{predicato}}(\underbrace{\underbrace{Amelia}_{\text{costante}}}_{\text{termine}},\ \underbrace{\underbrace{Alba}_{\text{costante}}}_{\text{termine}})}_{\text{formula atomica}}$$

$$\underbrace{\underbrace{shapeOf(\underbrace{b}_{\text{costante}})}_{\text{termine}} \underbrace{=}_{\text{predicato}} \underbrace{shapeOf(\underbrace{x}_{\text{variabile}})}_{\text{termine}}}_{\text{formula atomica}}$$

Regole di buona formazione

- Formule ben formate di L_{PO}

Ogni *formula atomica* è una fbf (poco diversa rispetto ad L_P)

$\varphi \in \text{fbf}(L_{PO}) \;\Rightarrow\; (\neg\varphi) \in \text{fbf}(L_{PO})$
$\varphi, \psi \in \text{fbf}(L_{PO}) \;\Rightarrow\; (\varphi \rightarrow \psi) \in \text{fbf}(L_{PO})$
$\varphi \in \text{fbf}(L_{PO}) \;\Rightarrow\; (\forall x\, \varphi) \in \text{fbf}(L_{PO})$ (aggiuntiva rispetto ad L_P)

$\varphi, \psi \in \text{fbf}(L_{PO})$, $(\varphi \vee \psi) \;\Leftrightarrow\; ((\neg\varphi) \rightarrow \psi)$
$\varphi, \psi \in \text{fbf}(L_{PO})$, $(\varphi \wedge \psi) \;\Leftrightarrow\; (\neg(\varphi \rightarrow (\neg\psi)))$
$\varphi, \psi \in \text{fbf}(L_{PO})$, $(\varphi \leftrightarrow \psi) \;\Leftrightarrow\; ((\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi))$
$\varphi \in \text{fbf}(L_{PO})$ $(\exists x\, \varphi) \Leftrightarrow (\neg\forall x\, \neg\varphi)$ (aggiuntiva rispetto ad L_P)

Formule aperte, enunciati

- Variabili **libere** e **vincolate**

  una variabile (in una fbf) è **vincolata** se si trova nel raggio di azione di un **quantificatore** per quella variabile

  una variabile è **libera** se non è *vincolata*

  esempi di variabile vincolata:
  $\forall x\ P(x)$
  $\exists x\ (P(x) \rightarrow (A(x) \wedge B(x))$

  esempi di variabile libera:
  $P(\boldsymbol{x})$
  $\exists y\ (P(y) \rightarrow (A(\boldsymbol{x},y) \wedge B(y)))$

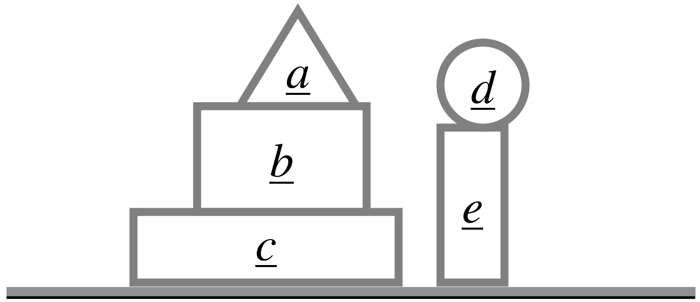
- **Formule aperte** e **chiuse**

  Una fbf è **aperta** se in essa vi è almeno una variabile libera

  Una fbf è **chiusa** (anche **enunciato** - *sentence*) in caso contrario

  solo le fbf *chiuse*, cioè gli *enunciati*, hanno un valore di verità (in quanto rappresentano delle *affermazioni* ...)

Semantica estensionale



- **Universo del discorso $\mathbf{U}$**

  Un insieme di oggetti di base

  $\{\underline{a}, \underline{b}, \underline{c}, \underline{d}, \underline{e}\}$

  Le costanti individuali indicano oggetti di $\mathbf{U}$

  Esempi: $v(a) = \underline{a}$, $v(b) = \underline{b}$

Qui si usa a per indicare la costante
ed $\underline{a}$ per indicare l'oggetto
(*solo per comodità)

- **Relazioni**

  Insiemi di tuple formate a partire da $\mathbf{U}$

  $\underline{Ontable} : \{\underline{c}, \underline{e}\}$

  $\underline{Above} : \{<\underline{a}, \underline{b}>, <\underline{a}, \underline{c}>, <\underline{b}, \underline{c}>, <\underline{d}, \underline{e}>\}$

  $\underline{On} : \{<\underline{a}, \underline{b}>, <\underline{b}, \underline{c}>, <\underline{d}, \underline{e}>\}$

  $\underline{Between} : \{<\underline{b}, \underline{a}, \underline{c}>\}$

  Predicati e funzioni indicano relazioni e funzioni in $\mathbf{U}$

  $v(Above(..)) : \{<\underline{a}, \underline{b}>, <\underline{a}, \underline{c}>, <\underline{b}, \underline{c}>, <\underline{d}, \underline{e}>\}$

Semantica

Strutture, interpretazioni e assegnazioni

- Una **struttura** <**U**, v> per L_{PO} contiene:

  Un insieme di oggetti **U** (l'universo del discorso)

  Un'**interpretazione** v che associa

  ad ogni **costante** c un **oggetto** di **U**
  $v(c) \in \mathbf{U}$

  ad ogni **predicato** P a n argomenti una **relazione** n-aria in $\mathbf{U}^n$
  $v(P) \subseteq \mathbf{U}^n$

  ad ogni **funzione** f a n argomenti una **funzione** da $\mathbf{U}^n$ a **U**
  $v(f) \subseteq \mathbf{U}^n \to \mathbf{U}$

  Notare che un'interpretazione non associa un valore delle *variabili*

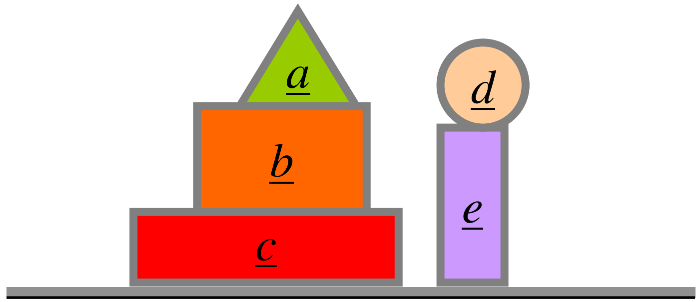
- Assegnazione

  Data una struttura <**U**, v>, un'**assegnazione** (*valuation*) s

  è una *funzione* che associa ad ogni **variabile** x un **oggetto** di **U**
  $s(x) \in \mathbf{U}$

  La combinazione di una <**U**, v> e di una s determina univocamente gli oggetti associati a ciascun elemento di L_{PO}

Il mondo dei blocchi



- Linguaggio

  predicati: $Ontable/1$, $Above/2$, $On/2$, $Between/3$, ...
  funzioni: $colorOf/1$, ...
  variabili: x, y, z, ...
  costanti individuali: a, b, c, d, e

- Interpretazione

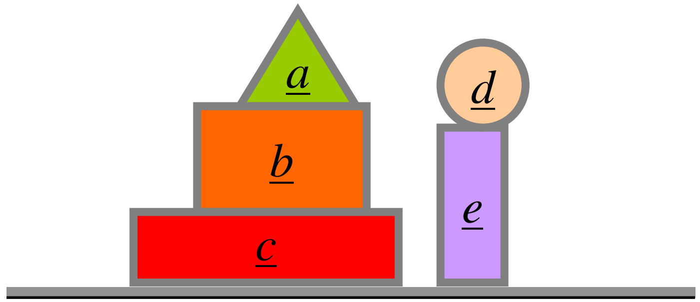
  Universo del discorso
  $\mathbf{U} = \{\underline{a}, \underline{b}, \underline{c}, \underline{d}, \underline{e}, \underline{red}, \underline{green}, \underline{violet}, \underline{pink}, \underline{orange}, \underline{nil}, \}$

  Predicati

  $v(Ontable/1) = \{\underline{c}, \underline{e}\}$
  $v(Above/2) = \{<\underline{a}, \underline{b}>, <\underline{a}, \underline{c}>, <\underline{b}, \underline{c}>, <\underline{d}, \underline{e}>\}$
  $v(On/2) = \{<\underline{a}, \underline{b}>, <\underline{b}, \underline{c}>, <\underline{d}, \underline{e}>\}$
  $\mathrm{v}(Between/3) = \{<\underline{b}, \underline{a}, \underline{c}>\}$

Si usa a per indicare la costante ed $\underline{a}$ per indicare l'oggetto (*solo per comodità)

Il mondo dei blocchi



- Linguaggio

  predicati: $Ontable/1$, $Above/2$, $On/2$, $Between/3$, ...
  funzioni: $colorOf/1$, ...
  variabili: x, y, z, ...
  costanti individuali: a, b, c, d, e

- Interpretazione

  Universo del discorso
  $\mathbf{U} = \{\underline{a}, \underline{b}, \underline{c}, \underline{d}, \underline{e}, \underline{red}, \underline{green}, \underline{violet}, \underline{pink}, \underline{orange}, \underline{nil}, \}$

  Si usa a per indicare la costante ed $\underline{a}$ per indicare l'oggetto (*solo per comodità)

  Funzioni
  $v(colorOf/1) = \{<\underline{a}, \underline{green}>, <\underline{b}, \underline{orange}>, <\underline{c}, \underline{red}>, <\underline{d}, \underline{pink}> <\underline{e}, \underline{violet}>\}$

  Costanti individuali
  $v(a) = \underline{a}$, $v(b) = \underline{b}$, $v(c) = \underline{c}$, $v(d) = \underline{d}$, $v(e) = \underline{e}$

- Assegnazione

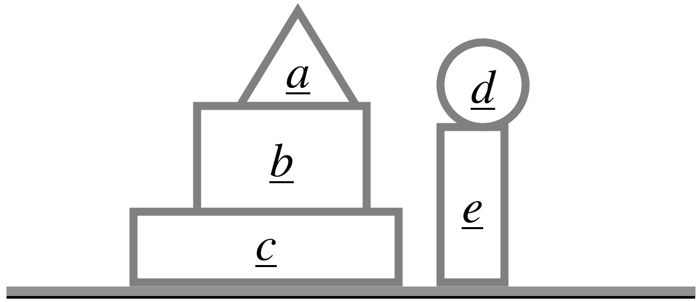
  Assegna un oggetto ad ogni variabile

  Esempio: $s = \{(x:\underline{a}), (y:\underline{b}), (z:\underline{a}) \ldots\}$

  (Un'assegnazione include *tutte* le variabili del linguaggio)

Soddisfacimento (forma intuitiva)

- Una fbf φ è soddisfatta da $<\mathbf{U}, v>$ $[s]$
  sse φ afferma una cosa vera in $<\mathbf{U}, v>$ $[s]$



- Nel mondo dei blocchi

  $Pyramid(a)$

  è vera perchè: $(v(a) = \underline{a}) \in v(Pyramid/1) = \{\underline{a}\}$

  $Parallelepiped(d)$

  non è vera perchè: $(v(d) = \underline{d}) \notin v(Parallelepiped/1) = \{\underline{b}, \underline{c}, \underline{e}\}$

  $\neg\exists x\ (Parallelepiped(x) \wedge Sphere(x))$

  è vera perché: $((v(Parallelepiped/1) = \{\underline{b}, \underline{c}, \underline{e}\}) \cap (v(Sphere/1) = \{\underline{d}\})) \equiv \varnothing$

  $\forall x\ (Pyramid(x) \vee Parallelepiped(x) \vee Sphere(x))$

  è vera perché:
  $((v(Pyramid/1) = \{\underline{a}\}) \cup (v(Parallelepiped/1) = \{\underline{b}, \underline{c}, \underline{e}\}) \cup (v(Sphere/1) = \{\underline{d}\})) \equiv \mathbf{U}$

Soddisfacimento

- Data una struttura $<\mathbf{U}, v>$ un'assegnazione s

  Se φ è una formula atomica, $<\mathbf{U}, v> [s] \models \varphi$ sse

  se φ ha la forma $P(t_1, \ldots , t_n)$ allora $<v(t_1) [s], \ldots, v(t_n) [s]> \in v(P) [s]$

  Se φ e ψ sono fbf qualsiasi

  Come per L_P

  $<\mathbf{U}, v> [s] \models (\neg\varphi)$ sse $<\mathbf{U}, v> [s] \not\models \varphi$

  $<\mathbf{U}, v> [s] \models (\varphi \wedge \psi)$ sse $<\mathbf{U}, v> [s] \models \varphi$ e $<\mathbf{U}, v> [s] \models \psi$

  $<\mathbf{U}, v> [s] \models (\varphi \vee \psi)$ sse $<\mathbf{U}, v> [s] \models \varphi$ o $<\mathbf{U}, v> [s] \models \psi$

  $<\mathbf{U}, v> [s] \models (\varphi \rightarrow \psi)$ allora non $<\mathbf{U}, v> [s] \models \varphi$ e $<\mathbf{U}, v> [s] \not\models \psi$

  Formule con quantificatori

  $<\mathbf{U}, v> [s] \models \forall x\, \varphi$ sse per ogni $\underline{d} \in \mathbf{U}$ si ha $<\mathbf{U}, v> [s](x{:}\underline{d}) \models \varphi$

  $<\mathbf{U}, v> [s] \models \exists x\, \varphi$ sse esiste un $\underline{d} \in \mathbf{U}$ per cui si ha $<\mathbf{U}, v> [s](x{:}\underline{d}) \models \varphi$

Modelli

- Validità in un’interpretazione, modello

  Una fbf φ tale per cui si ha $<\mathbf{U}, v> [s] \models \varphi$ per qualsiasi assegnazione s è detta valida in $<\mathbf{U}, v>$

  Si dice anche che $<\mathbf{U}, v>$ è un modello di φ

  si scrive $<\mathbf{U}, v> \models \varphi$ (si elimina il riferimento a s)

  Una struttura $<\mathbf{U}, v>$ è detta modello di un *insieme di fbf* Γ sse è un modello di tutte le fbf in Γ

  si scrive allora $<\mathbf{U}, v> \models \Gamma$

- Verità

  Un enunciato ψ si dice vero in $<\mathbf{U}, v>$ se è valido in $<\mathbf{U}, v>$

  per un enunciato, basta l’esistenza di un’assegnazione s per cui $<\mathbf{U}, v> [s] \models \psi$

Validità

- Validità e verità logiche

  Una fbf (aperta o chiusa) è **valida** (o **logicamente valida**) se è **valida** in qualsiasi $<\mathbf{U}, v>$

  Esempi:

  $(P(x) \vee \neg P(x))$ (tautologia tradotta in fbf aperta)

  Un enunciato ψ è **vero** (o **logicamente vero**) se è **vero** in qualsiasi $<\mathbf{U}, v>$

  si scrive allora $\models \psi$ (si elimina il riferimento a $<\mathbf{U}, v>$)

  Esempi:

  $\forall x\, (P(x) \vee \neg P(x))$ (generalizzazione di una tautologia)

  $\forall x \forall y\, (G(x,y) \rightarrow (H(x,y) \rightarrow G(x,y)))$ (generalizzazione di assioma – vedi oltre)

- Inconsistenza

  Una fbf (aperta o chiusa) è **inconsistente** se non è soddisfacibile

  Un enunciato ψ è **inconsistente** se non ha un *modello*

  Esempi:

  $\forall x\, (P(x) \wedge \neg P(x))$ (generalizzazione di una contraddizione)

Conseguenza logica

- Definizione

  Dato un insieme di fbf Γ ed una fbf φ di L_{PO} si ha

  $$\Gamma \models \varphi$$

  sse tutte le $<\mathbf{U}, v> [s]$ che soddisfano Γ soddisfano anche φ

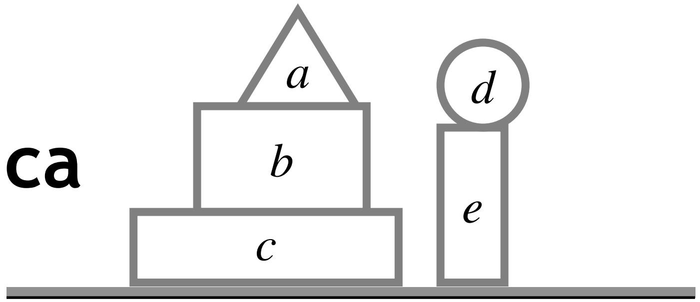
- Osservazioni

  La definizione si estende a tutte le possibili $<\mathbf{U}, v> [s]$

  Quindi, a tutti i possibili insiemi $\mathbf{U}$, alle relazioni e funzioni in $\mathbf{U}$ ed alle associazioni di oggetti di $\mathbf{U}$ a variabili e costanti

  Il calcolo diretto della conseguenza logica in L_{PO} è impossibile anche nelle forme più semplici

Assiomatizzazioni, conseguenza logica



- Assiomi del mondo dei blocchi (Cook & Liu, 2002)

  B1: $\forall x \forall y\ (On(x,y) \leftrightarrow (Above(x,y)\ \wedge\ \neg\exists z\ (Above(x,z)\ \wedge\ Above(z,y))))$

  B2: $\forall x\ (Ontable(x) \leftrightarrow \neg\exists z\ Above(x,z))$

  B3: $\forall x\ (Clear(x) \leftrightarrow \neg\exists z\ Above(z,x))$

  Tutte le verità del mondo dei blocchi sono una conseguenza logica di B1-B3

  Esempio (intuitivamente vero):

  $\forall x \forall y \forall z\ (On(x,y)\ \wedge\ On(y,z)) \rightarrow Above(x,z))$

  Vale in qualsiasi mondo dei blocchi

  Esempio:

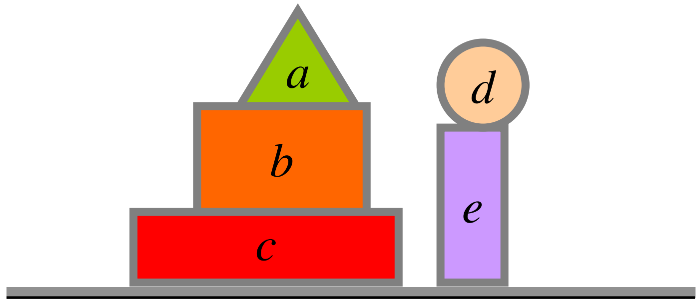
  $On(a,b)\ \wedge\ On(b,c)\ \wedge\ Above(a,b)$

  Vale nel mondo dei blocchi in figura

- In generale

  Il problema è come calcolare se $\Gamma \models \varphi$

A cosa servono le funzioni



- Le funzioni descrivono relazioni o proprietà

  Esempio:
  $(colorOf(a) = green) \wedge (colorOf(b) = orange) \wedge (colorOf(c) = red)$
  $(colorOf(d) = pink) \wedge (colorOf(e) = violet)$

  Esempio:
  $(searchDepth(s0) = 2) \wedge (parent(s0) = nil) \wedge (farmerLocation(s) = shore1) \wedge (foxLocation(s) = shore2))$

La comodità di avere *nil*

Tipicamente, le funzioni che esprimono proprietà non sono definite per tutti gli oggetti

Per completare la definizione di una funzione, si usa *nil* per esprimere un non-valore

Esempio:
$(colorOf(a) = green) \wedge (colorOf(green) = nil)$

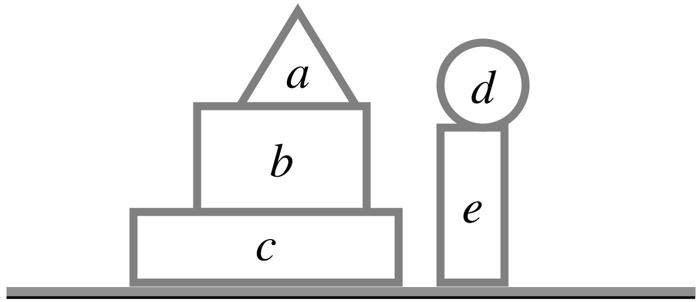
Un colore è inteso come un valore, non un oggetto. Quindi non ha un colore.

L'interpretazione di *nil* è intesa come un oggetto speciale, un non-oggetto

Esempio: interpretazione completa di $colorOf/1$

$v(colorOf/1) = \{<\underline{a}, \underline{green}>, <\underline{b}, \underline{orange}>, <\underline{c}, \underline{red}>, <\underline{d}, \underline{pink}>, <\underline{e}, \underline{violet}>, <\underline{green}, \underline{nil}>, <\underline{orange}, \underline{nil}>, <\underline{red}, \underline{nil}>, <\underline{pink}, \underline{nil}> <\underline{violet}, \underline{nil}>, <\underline{nil}, \underline{nil}> \}$

A cosa servono i predicati



- I predicati descrivono appartenenza a categorie

  Esempio:
  $Parallelepiped(b) \wedge Parallelepiped(c) \wedge Parallelepiped(e)$

- I predicati descrivono relazioni tra oggetti

  Esempio:
  $Above(a,b) \wedge Above(b,c) \wedge Above(a,c) \wedge Above(d,e)$

- Differenze tra predicati e funzioni

  Differenze sintattiche (*linguaggio*):
  Le funzioni si possono nidificare, i predicati no.

  Esempio:
  $\neg \forall x\, \exists y\, ((y = parent(parent(parent(x)))) \wedge (farmer\text{-}location(y) = shore1))$

  Non in tutti gli stati, tornando indietro di tre passi si incontra uno stato dove il *farmer* è in *shore1*

  Differenze semantiche (*significato*):
  Le funzioni identificano un oggetto, i predicati descrivono una relazione

  Esempi:
  $parent(s0)$ è un oggetto ben preciso
  $Above(a,b)$ descrive una relazione tra a e b

Predicati e funzioni insieme in L_{PO}

- Un esempio notevole

Assiomi di Peano (secondo Mendelson, 1972)

S1: $\forall x \forall y \forall z\ ((x = y) \rightarrow ((x = z) \rightarrow (y = z)))$

S2: $\forall x \forall y\ ((x = y) \rightarrow (s(x) = s(y)))$

S3: $\neg \exists x\ (s(x) = 0)$

S4: $\forall x \forall y\ ((s(x) = s(y)) \rightarrow (x = y))$

S5: $\forall x\ (x + 0 = x)$

S6: $\forall x \forall y\ ((x + s(y)) = s(x + y))$

S7: $\forall x\ (x \cdot 1 = x)$

S8: $\forall x \forall y\ ((x \cdot s(y)) = ((x \cdot y) + x))$

S9: *Per qualsiasi fbf* $\varphi(x)$: $\varphi(0) \rightarrow (\forall x\ (\varphi(x) \rightarrow \varphi(s(x)))) \rightarrow \forall x\ \varphi(x)$

(~*principio di induzione matematica*)

Un solo schema d'assioma: **S9**

Tre predicati: $=/2$, $+/2$, $\cdot/2$

Una funzione: $s/1$

Tutte le proprietà dei numeri interi sono conseguenza logica di **S1-S9**

FAQ

- Funzioni o predicati (i.e. relazioni)?

  I due oggetti semantici sono molto simili, si può fare a meno delle funzioni?

- Le funzioni (come oggetti semantici) si possono *rappresentare* anche tramite predicati

  ad esempio, la validità dell'enunciato:

  $$\forall x \forall y \forall z\ ((\varphi(x,y) \wedge \varphi(x,z)) \rightarrow (y = z))$$

  indica che l'interpretazione di $\varphi(..)$ (in generale, una relazione $v(\varphi) \subseteq \mathbf{U}^2$) è anche una funzione $\mathbf{U} \rightarrow \mathbf{U}$

  La presenza delle funzioni arricchisce il linguaggio in modo sostanziale: a differenza dei predicati, le funzioni si possono nidificare (nei termini)

  Di conseguenza, viene grandemente aumentata la portata del calcolo logico-simbolico

  (*con un corrispondente aumento della complessità di calcolo ...*)

Automazione (L_{PO} in forma ristretta e forward chaining)

L_{PO} in forma ristretta

- Nelle fbf di L_{PO} considerate da questo punto in poi
  - Non compaiono simboli funzionali (solo costanti e variabili)
  - Compare solo il quantificatore universale, in forma positiva
    - Il linguaggio contiene un numero finito di costanti individuali
  - Esempi:
    - $\forall x\, \forall y\, (P(x) \rightarrow (Q(x, y) \wedge R(y)))$
    - $\forall x\, (P(x)) \rightarrow \forall y\, \forall z\, ((Q(y, z) \wedge R(z)))$
    - $\forall x\, (P(x)) \rightarrow \neg \forall z\, (R(z))$ (*NO: equivale a $\forall x\, (P(x)) \rightarrow \exists z\, \neg (R(z))$)

- Forma normale prenessa
  - Le fbf di sono traducibili in una forma
    - $\forall x_1 \forall x_2 \ldots \forall x_n \varphi$
    - dove φ non contiene quantificatori (φ è anche detta *matrice*)
  - Procedura
    - Si spostano i quantificatori all'inizio, eventualmente ridenominando le variabili
      - $\forall x\, (P(x)) \rightarrow \forall x\, (R(x))$ diventa $\forall x\, \forall y\, (P(x) \rightarrow R(y))$

Forma a clausole in L_{PO}

- Enunciati in forma universalmente quantificata

  Si parte dalla fbf in forma prenessa

  $\forall x_1 \forall x_2 \ldots \forall x_n \varphi$ (φ non contiene quantificatori)

  Essendo tutti universali, i quantificatori si possono semplicemente omettere

  La matrice φ viene tradotta in FNC (con le stesse regole del caso proposizionale)
  e quindi in forma a clausole FC (con le stesse regole del caso proposizionale)

  Esempio:

  $\forall x \, \forall y \, (P(x) \rightarrow (Q(x, y) \wedge R(y)))$ (fbf di partenza)

  $P(x) \rightarrow (Q(x, y) \wedge R(y))$ (eliminazione dei quantificatori)

  $\neg P(x) \vee (Q(x, y) \wedge R(y))$ (equivalenza di $\rightarrow$)

  $(\neg P(x) \vee Q(x, y)) \wedge (\neg P(x) \vee R(y))$ (FNC, distribuitività di $\vee$)

  $\{\neg P(x), Q(x, y)\}, \ \{\neg P(x), R(y)\}$ (FC)

  Si dicono base (*ground*) le clausole che non contengono variabili

Clausole di Horn in L_{PO}

- Definizione quasi identica al caso proposizionale
  - Forma a clausole (FC) di enunciati universalmente quantificati
  - In ciascuna clausola occorre al massimo un letterale in forma positiva

- Fatti, regole e goal
  - **Fatti**: clausola con un singolo atomo in forma positiva

    $\{Umano(socrate)\}, \{Pyramid(a)\}, \{Sorella(alba, paolo)\}$
  - **Regole**: clausola di due o più atomi, uno in forma positiva

    $\{Umano(x), \neg Filosofo(x)\},$

    $\forall x\ (Filosofo(x) \rightarrow Umano(x))$

    $\{Madre(x,y), \neg Femmina(x), \neg Genitore(x, y)\}$

    $\forall x \forall y\ ((Femmina(x) \land Genitore(x, y)) \rightarrow Madre(x, y))$
  - **Goal**: clausola di atomi in forma negativa

    $\{\neg Umano(socrate)\}$

    $\{\neg Sorella(alba, amelia)\}$

Base (finita) di Herbrand

- Atomi di Herbrand in L_{PO}

  Un atomo di Herbrand è una fbf *atomica* (*ground atom*) che non contiene variabili (solo costanti individuali)

  Esempi: $P(a)$, $P(d)$, $P(e)$, $Q(a, b)$, $Q(d, e)$, $R(a, c, e)$, ...

- Base (finita) di Herbrand

  La base di Herbrand è l'insieme di tutti gli atomi di Herbrand

  $\mathrm{B_H} \equiv \{P(a), P(b), P(c), ..., Q(a, a), Q(a, b), ..., R(a, a, a), R(a, a, b), R(a, a, c), ...\}$

  Se il linguaggio ha un numero finito di predicati e costanti, la base di Herbrand è finita

Modelli (finiti) di Herbrand

- **Struttura** (finita) di Herbrand per L_{PO}

  Una struttura $<\mathbf{U}_H, v_H>$ tale che

  $\mathbf{U}_H$ è formato dalle costanti individuali del linguaggio ($\mathrm{Cost}(L_{PO})$)

  v_H di fatto, è un sottoinsieme della base di Herbrand

- **Interpretazione** v_H di Herbrand

  $v_H \equiv \{P(a), P(b), P(c), Q(a, b), R(a, c) \ldots \}$ (solo formule *ground atoms*)

  $v_H \subseteq \mathrm{B}_H$

  (Tramite la base di Herbrand B_H si interpreta un linguaggio usando come universo gli oggetti del linguaggio stesso …)

- **Modello** di Herbrand

  $\varphi \in \mathrm{Atomi}(L_{PO})$, $<\mathbf{U}_H, v_H> [s] \models \varphi$ sse $\varphi \in v_H$

  $\varphi \in \mathrm{Atomi}(L_{PO})$, $<\mathbf{U}_H, v_H> [s] \models \neg\varphi$ sse $\varphi \notin v_H$

  $<\mathbf{U}_H, v_H> [s] \models \neg\varphi$ sse $<\mathbf{U}_H, v_H> [s] \not\models \varphi$

  $<\mathbf{U}_H, v_H> [s] \models \varphi \rightarrow \psi$ sse non $(<\mathbf{U}_H, v_H> [s] \models \varphi$ e $<\mathbf{U}_H, v_H> [s] \not\models \psi)$

  $<\mathbf{U}_H, v_H> [s] \models \forall x\, \varphi$ se per ogni $c \in \mathrm{Cost}(L_{PO})$ si ha $<\mathbf{U}_H, v_H> [s](x{:}c) \models \varphi$

Sistema di Herbrand (in forma ristretta)

- **Sistema** (finito) di Herbrand per un enunciato

  Dato un enunciato universale, cioè della forma

  $\forall x_1 \forall x_2 \dots \forall x_n \varphi$ (φ non contiene quantificatori)

  è l'insieme (finito) di formule chiuse generato per sostituzione

  $\varphi\ [x_1/c_1, x_2/c_2 \dots x_n/c_n]$

  con <u>tutte</u> le possibili combinazioni $<c_1, c_2 \dots c_n>$ di $c_i \in \mathrm{Cost}(L_{PO})$

- **Sistema** (finito) di Herbrand di una teoria

  Data una teoria Σ di enunciati universali, il sistema (finito) $\mathrm{H}(\Sigma)$
  è l'unione di tutti i sistemi di Herbrand generati dagli enunciati Σ

Teorema di Herbrand (in forma ristretta)

- **Teorema di Herbrand** (in forma ristretta)

  Data una teoria di enunciati universali Γ,
  $H(\Gamma)$ ha un modello sse Γ ha un modello

- Corollario

  Sia Γ un insieme di clausole di Horn, le seguenti affermazioni sono equivalenti:
  - Γ è soddisfacibile
  - Γ ha un modello di Herbrand

  Non vale in generale: solo se Γ è un insieme clausole di Horn

  In questa forma (finita), è quasi una procedura effettiva ...

Derivazioni: sistema di Herbrand

- Esempio:

$\Gamma \equiv \{\{Umano(x), \neg Filosofo(x)\}, \{Mortale(y), \neg Umano(y)\}\}$
$\text{Cost}(L_{PO}) \equiv \{socrate, platone, aristotele\}$

Sistema di Herbrand di Γ

$\text{H}(\Gamma) \equiv \{\{Umano(socrate), \neg Filosofo(socrate)\}, \{Umano(platone), \neg Filosofo(platone)\},$
$\{Umano(aristotele), \neg Filosofo(aristotele)\}, \{Mortale(socrate), \neg Umano(socrate)\},$
$\{Mortale(platone), \neg Umano(platone)\}, \{Mortale(aristotele), \neg Umano(aristotele)\}\}$

Aggiungiamo un insieme di fatti

$\Sigma \equiv \{Filosofo(socrate)\}, \{Filosofo(platone)\}, \{Filosofo(aristotele)\}\}$

Applicando il *Modus Ponens* in modo esaustivo a $\text{H}(\Gamma) \cup \Sigma$:

$\text{K} \equiv \{\{Umano(socrate)\}, \{Umano(platone)\}, \{Umano(aristotele)\},$
$\{Mortale(socrate)\}, \{Mortale(platone)\}, \{Mortale(aristotele)\}\}$

Regola di istanziazione

- Una regola di inferenza specifica (per L_{PO}):

$$INST \qquad \frac{\forall x_1 \forall x_2 \ldots \forall x_n \varphi}{\varphi\, [x_1/c_1, x_2/c_2 \ldots x_n/c_n]}$$

Da un enunciato universalmente quantificato si può derivare un enunciato (non quantificato) in cui tutte le variabili sono sostituite da costanti

Esempi:

$\forall x\, (Filosofo(x) \rightarrow Umano(x)) \;\vdash\; Filosofo(socrate) \rightarrow Umano(socrate)$

$\forall x \forall y\, ((Femmina(x) \land Genitore(x, y)) \rightarrow Madre(x, y)) \vdash$
$((Femmina(paola) \land Genitore(paola, amelia)) \rightarrow Madre(paola, amelia))$

La regola *INST* è **corretta** (in L_{PO})

Il sistema di Herbrand di un enunciato universalmente quantificato si ottiene applicando esaustivamente *INST* per tutte le possibili sostituzioni $x_1/c_1, x_2/c_2 \ldots x_n/c_n$

Metodo del forward chaining in L_{PO}

- Descrizione:

Per stabilire se $\Gamma \vdash \varphi$

(Γ regole e fatti come clausole definite universalmente quantificate, φ fatto)

Si applicano le regole *INST* e *GMP* a Γ in modo esaustivo

INST: $\forall x_1 \forall x_2 \ldots \forall x_n \varphi \vdash \varphi\, [x_1/c_1, x_2/c_2 \ldots x_n/c_n]$

GMP: $\alpha_1,\ \alpha_2,\ \ldots\ \alpha_n, \alpha_1 \land \alpha_2 \land \ldots \land \alpha_n \rightarrow \beta \vdash \beta$

($\alpha_1,\ \alpha_2,\ \ldots\ \alpha_n, \beta$ sono fbf *ground*, vale a dire dove non occorrono variabili)

Si ottiene un insieme Σ di fatti, $\Sigma = \{\psi : \Gamma \vdash \psi, \textit{per INST e GMP}\}$

L'algoritmo termina con successo se $\varphi \in \Sigma$

Esempio:

INST: $\forall x\, \forall y\, ((Femmina(x) \land Genitore(x, y)) \rightarrow Madre(x, y)) \vdash$
$((Femmina(paola) \land Genitore(paola, amelia)) \rightarrow Madre(paola, amelia))$

GMP: $Femmina(paola), Genitore(paola, amelia),$
$((Femmina(paola) \land Genitore(paola, amelia)) \rightarrow Madre(paola, amelia)) \vdash$
$Madre(paola, amelia)$

Caratteristiche del forward chaining in L_{PO}

- Il metodo del forward chaining per le clausole di Horn in L_{PO} (in forma ristretta)
  - Termina sempre (se le costanti del linguaggio sono in numero finito)
  - E` corretto
  - E` completo
    - Attenzione: tutto ciò vale solo per le clausole di Horn enunciati univer
- Procedura effettiva:
  - Per la regola GMP, non è particolarmente problematica
  - Per la regola $INST$, si dovrebbero generare tutte le possibili *istanziazioni*
    - Altamente inefficiente, comunque: ciascuna fbf quantificata produce n^m fbf *ground* (n numero delle costanti, m numero delle variabili quantificate)
- Miglioramenti per $INST$
  - Se l'obiettivo è applicare GMP a regole e fatti *ground* (privi di variabili)
  - Ci si potrebbe limitare alla generazione delle sole istanziazioni delle regole cui è applicabile GMP, dati i fatti in Γ
  - Ma come?

Sistemi a regole (Jess)

- Un sistema a regole contiene regole e fatti
    - Nelle regole, tutte le variabili sono universalmente quantificate
    - I fatti non contengono variabili
- Ciascuna regola è un'implicazione
    - <LHS - *Left Hand Side*> $\Rightarrow$ <RHS - *Right Hand Side*>
    - LHS e RHS
        - Sono congiunzioni di fbf atomiche in forma positiva
        - Morale: le regole Jess sono traducibili in regole di Horn

Regola "Sorella"

$Madre(m, x)$
$Madre(m, y)$
$Padre(p, x)$
$Padre(p, y)$
$Femmina(x)$
$\Rightarrow$
$Sorella(x, y)$

Regola "Fratello"

$Madre(m, x)$
$Madre(m, y)$
$Padre(p, x)$
$Padre(p, y)$
$Maschio(x)$
$\Rightarrow$
$Fratello(x, y)$