Logica del Primo Ordine

Corso di Sistemi Intelligenti, Capitolo 8

Outline

- ♢ Perchè FOL (First-Order Logic) ?
- ♢ Sintassi e Semantica di FOL
- ♢ Esempi di sentenze
- ♢ Il mondo dei Wumpus in FOL

Pro e contro della Logica Proposizionale

 La Logica Proposizionale è *dichiarativa*: pezzi di sintassi corrispondono a fatti

 La Logica Proposizionale permette di esprimere informazione parziale/disgiuntiva/negata
(al contrario di molte strutture dati e basi di dati)

 La Logica Proposizionale è *composizionale*:
il significato di $B_{1,1} \wedge P_{1,2}$ è derivato dal significato di $B_{1,1}$ e di $P_{1,2}$

 Il significato nella Logica Proposizionale è *indipendente dal contesto*
(al contrario del linguaggio naturale, dove il significato dipende dal contesto)

 La Logica Proposizionale ha una potenza espressiva molto limitata
(al contrario del linguaggio naturale)
P.e., non si può dire "le trappole causano la brezza in quadrati adiacenti"
se non scrivendo una sentenza per quadrato

Logica del Primo Ordine (First-order logic)

Mentre la Logica Proposizionale assume che il mondo contenga *fatti*,
la Logica del Primo Ordine (come il linguaggio naturale) assume che il mondo contenga

- Oggetti: persone, case, numeri, teorie, Paolino Paperino, colori, partite di calcio, guerre, secoli ...
- Relazioni: rosso, rotondo, finto, primo ..., fratello di, più grande di, dentro, parte di, ha colore, occorso dopo, possiede, ...
- Funzioni: padre di, miglior amico, secondo tempo di, inizio di ...

Logica in generale

Linguaggio	Scelta Ontologica	Scelta Epistemologica
Logica Proposizionale	fatti	vero/falso/sconosciuto
Logica del Primo Ordine	fatti, oggetti, relazioni	vero/falso/sconosciuto
Logica Temporale	fatti, oggetti, relazioni, tempi	vero/falso/sconosciuto
Teoria delle Probabilità	fatti	gradi di credenza $\in [0, 1]$
Logica Fuzzy	gradi di verità $\in [0, 1]$	intervalli di valore conosciuti

Sintassi di FOL: elementi base

Costanti	$ReGiovanni,\ 2,\ UP, \ldots$
Predicati	$Fratello,\ >, \ldots$
Funzioni	$Sqrt,\ GambaSinistraDi, \ldots$
Variabili	$x,\ y,\ a,\ b, \ldots$
Connettivi	$\wedge\ \vee\ \neg\ \Rightarrow\ \Leftrightarrow$
Uguaglianza	$=$
Quantificatori	$\forall\ \exists$

Sentenze Atomiche

Sentenza atomica $= predicato(termine_1, \ldots, termine_n)$
o $termine_1 = termine_2$

Termine $= funzione(termine_1, \ldots, termine_n)$
o $costante$ o $variabile$

P.e., $Fratello(ReGiovanni, RiccardoCuorDiLeone)$
$> (Lung(GambaSinistraDi(Riccardo)), Lung(GambaSinistraDi(ReGiovanni)))$

Sentenze Complesse

Le sentenze complesse sono create da sentenze atomiche usando i connettivi

$$\neg S, \quad S_1 \wedge S_2, \quad S_1 \vee S_2, \quad S_1 \Rightarrow S_2, \quad S_1 \Leftrightarrow S_2$$

P.e. $Fratello(ReGiovanni, Riccardo) \Rightarrow Fratello(Riccardo, ReGiovanni)$

$>(1,2) \vee \leq(1,2)$

$>(1,2) \wedge \neg>(1,2)$

Verità nella Logica del Primo Ordine

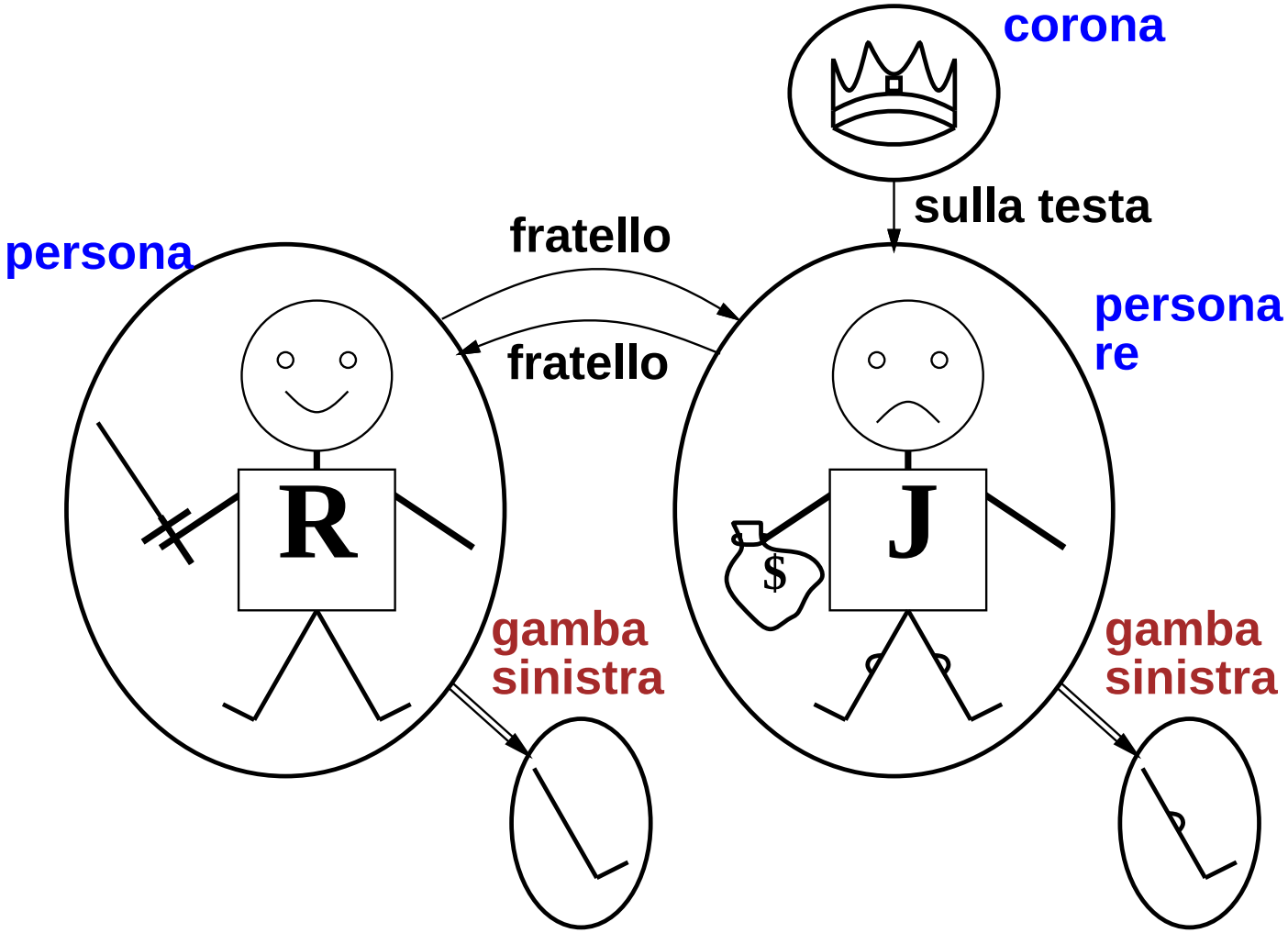
Le sentenze sono vere rispetto ad un modello ed una interpretazione

Il modello contiene ≥ 1 oggetti (elementi di dominio) e relazioni fra loro

L'interpretazione specifica i referenti per
- *simboli di costanti* $\rightarrow$ oggetti
- *simboli di predicati* $\rightarrow$ relazioni
- *simboli di funzione* $\rightarrow$ relazioni funzionali

Una sentenza atomica $predicato(termine_1, \ldots, termine_n)$ è vera
sse gli oggetti riferiti da $termine_1, \ldots, termine_n$
sono nella relazione riferita dal $predicato$

Modelli per FOL: Esempio



Modelli per FOL: Tanti!

Si *possono* enumerare i modelli per il vocabolario di una data KB:

Per ogni numero di elementi di dominio n, da 1 a ∞
    Per ogni predicato k-ario P_k nel vocabolario
        Per ogni possibile relazione k-aria su n oggetti
            Per ogni simbolo di costante C nel vocabolario
                Per ogni scelta di referente per C da n oggetti ...

Calcolare le conseguenze logiche enumerando i modelli non è facile!

Quantificatori Universali

$\forall \langle variabili \rangle \ \langle sentenza \rangle$

Chiunque a Padova è intelligente:
$\forall x \ \ Luogo(x, Padova) \ \Rightarrow \ Intelligente(x)$

$\forall x \ \ P$ è vero in un modello m sse P è vero essendo x
ogni possibile oggetto nel modello

In prima approssimazione, equivalente alla congiunzione di istanziazioni di P

$$\begin{aligned} & Luogo(ReGiovanni, Padova) \ \Rightarrow \ Intelligente(ReGiovanni) \\ \wedge \ & Luogo(Riccardo, Padova) \ \Rightarrow \ Intelligente(Riccardo) \\ \wedge \ & Luogo(Padova, Padova) \ \Rightarrow \ Intelligente(Padova) \\ \wedge \ & \ldots \end{aligned}$$

Uno sbaglio comune da evitare

Tipicamente, $\Rightarrow$ è il connettivo principale usato con $\forall$

Sbaglio comune: uso di $\wedge$ come connettivo principale con $\forall$:

$$\forall x \;\; Luogo(x, Padova) \wedge Intelligente(x)$$

significa "Chiunque è a Padova e chiunque è intelligente"

Quantificatore esistenziale

$\exists \langle variabili \rangle \ \ \langle sentenza \rangle$

Qualcuno a Bologna è intelligente:
$\exists x \ \ Luogo(x, Bologna) \wedge Intelligente(x)$

$\exists x \ \ P$ è vero in un modello m sse P è vero essendo x
qualche possibile oggetto nel modello

In prima approssimazione, equivalente alla disgiunzione di istanziazioni di P

$$\begin{array}{ll} & Luogo(ReGiovanni, Bologna) \wedge Intelligente(ReGiovanni) \\ \vee & Luogo(Richard, Bologna) \wedge Intelligente(Richard) \\ \vee & Luogo(Bologna, Bologna) \wedge Intelligente(Bologna) \\ \vee & \ldots \end{array}$$

Un altro sbaglio comune da evitare

Tipicamente, $\wedge$ è il connettivo principale usato con $\exists$

Sbaglio comune: uso di $\Rightarrow$ come connettivo principale con $\exists$:

$$\exists x \;\; Luogo(x, Bologna) \;\Rightarrow\; Intelligente(x)$$

è vero se c'è qualcuno che non è a Bologna!

Proprietà dei quantificatori

$\forall x \ \forall y$ è lo stesso di $\forall y \ \forall x$ (Perchè??)

$\exists x \ \exists y$ è lo stesso di $\exists y \ \exists x$ (Perchè??)

$\exists x \ \forall y$ non è lo stesso di $\forall y \ \exists x$

$\exists x \ \forall y \ \ Ama(x, y)$
"C'è una persona che ama chiunque nel mondo"

$\forall y \ \exists x \ \ Ama(x, y)$
"Ognuno nel mondo è amato da almeno una persona"

Dualità dei quantificatori: ognuno può essere espresso usando l'altro

$\forall x \ \ Piace(x, gelato) \qquad \neg\exists x \ \neg Piace(x, gelato)$

$\exists x \ \ Piace(x, broccoli) \qquad \neg\forall x \ \neg Piace(x, broccoli)$

Esempi di sentenze

Fratelli sono siblings (fratelli germani: leggi "fratelli e sorelle")

Esempi di sentenze

Fratelli sono siblings

$\forall\, x, y \;\; Fratello(x, y) \;\Rightarrow\; Sibling(x, y).$

"Sibling" è simmetrico

Esempi di sentenze

Fratelli sono siblings

$\forall\, x, y \;\; Fratello(x, y) \;\Rightarrow\; Sibling(x, y).$

"Sibling" è simmetrico

$\forall\, x, y \;\; Sibling(x, y) \;\Leftrightarrow\; Sibling(y, x).$

La madre di una persona è il genitore femmina della persona

Esempi di sentenze

Fratelli sono siblings

$\forall\, x, y \;\; Fratello(x, y) \;\Rightarrow\; Sibling(x, y).$

"Sibling" è simmetrico

$\forall\, x, y \;\; Sibling(x, y) \;\Leftrightarrow\; Sibling(y, x).$

La madre di una persona è il genitore femmina della persona

$\forall\, x, y \;\; Madre(x, y) \;\Leftrightarrow\; (Femmina(x) \wedge Genitore(x, y)).$

Un cugino di primo grado è un figlio di un "sibling" di un genitore

Esempi di sentenze

Fratelli sono siblings

$\forall x, y \;\; Fratello(x, y) \;\Rightarrow\; Sibling(x, y).$

"Sibling" è simmetrico

$\forall x, y \;\; Sibling(x, y) \;\Leftrightarrow\; Sibling(y, x).$

La madre di una persona è il genitore femmina della persona

$\forall x, y \;\; Madre(x, y) \;\Leftrightarrow\; (Femmina(x) \wedge Genitore(x, y)).$

Un cugino di primo grado è un figlio di un "sibling" di un genitore

$\forall x, y \;\; CuginoIGrado(x, y) \;\Leftrightarrow\; \exists p, ps \;\; Genitore(p, x) \wedge Sibling(ps, p) \wedge Genitore(ps, y)$

Uguaglianza

$termine_1 = termine_2$ è vero per una data interpretazione
se e solo se $termine_1$ e $termine_2$ si riferiscono allo stesso oggetto

P.e., $1 = 2$ e $\forall x \;\; \times(Sqrt(x), Sqrt(x)) = x$ sono soddisfacibili
$2 = 2$ è valido

P.e., definizione di $Sibling$ in termini di $Genitore$:

$$\forall x, y \;\; Sibling(x, y) \;\Leftrightarrow\; [\neg(x = y) \wedge \exists m, f \;\; \neg(m = f) \wedge Genitore(m, x) \wedge Genitore(f, x) \wedge Genitore(m, y) \wedge Genitore(f, y)]$$

Assiomatizzazione del concetto di Insieme

Simbolo costante: $\{\}$ — insieme vuoto

Assiomatizzazione del concetto di Insieme

Simbolo costante: $\{\}$ — insieme vuoto

Predicato unario: $Set(s)$ — insieme s

Assiomatizzazione del concetto di Insieme

Simbolo costante: $\{\}$ — insieme vuoto

Predicato unario: $Set(s)$ — insieme s

Predicato binario: $x \in s$ — scrittura infissa per $\in (x, s)$, x è un membro di s

Predicato binario: $s_1 \subseteq s_2$ — s_1 è un sottoinsieme (non necess. proprio) di s_2

Assiomatizzazione del concetto di Insieme

Simbolo costante: $\{\}$ — insieme vuoto

Predicato unario: $Set(s)$ — insieme s

Predicato binario: $x \in s$ — scrittura infissa per $\in (x, s)$, x è un membro di s

Predicato binario: $s_1 \subseteq s_2$ — s_1 è un sottoinsieme (non necess. proprio) di s_2

Funzione binaria: $s_1 \cup s_2$ — unione di s_1 e s_2

Funzione binaria: $s_1 \cap s_2$ — intersezione di s_1 e s_2

Funzione binaria: $\{x|s_2\}$ — insieme risultante dall'aggiungere x all'insieme s

Assiomatizzazione del concetto di Insieme

Simbolo costante: $\{\}$ — insieme vuoto
Predicato unario: $Set(s)$ — insieme s
Predicato binario: $x \in s$ — scrittura infissa per $\in (x, s)$, x è un membro di s
Predicato binario: $s_1 \subseteq s_2$ — s_1 è un sottoinsieme (non necess. proprio) di s_2
Funzione binaria: $s_1 \cup s_2$ — unione di s_1 e s_2
Funzione binaria: $s_1 \cap s_2$ — intersezione di s_1 e s_2
Funzione binaria: $\{x|s_2\}$ — insieme risultante dall'aggiungere x all'insieme s

$$\forall\, s \;\; Set(s) \;\Leftrightarrow\; (s = \{\}) \vee (\exists\, x, s_2 \;\; Set(s_2) \wedge s = \{x|s_2\})$$

I soli insiemi sono quello vuoto o quelli ottenuti aggiungendo qualcosa ad un insieme

Assiomatizzazione del concetto di Insieme

Simbolo costante: $\{\}$ — insieme vuoto
Predicato unario: $Set(s)$ — insieme s
Predicato binario: $x \in s$ — scrittura infissa per $\in (x, s)$, x è un membro di s
Predicato binario: $s_1 \subseteq s_2$ — s_1 è un sottoinsieme (non necess. proprio) di s_2
Funzione binaria: $s_1 \cup s_2$ — unione di s_1 e s_2
Funzione binaria: $s_1 \cap s_2$ — intersezione di s_1 e s_2
Funzione binaria: $\{x|s_2\}$ — insieme risultante dall'aggiungere x all'insieme s

$\forall\, s \;\; Set(s) \;\Leftrightarrow\; (s = \{\}) \vee (\exists\, x, s_2 \;\; Set(s_2) \wedge s = \{x|s_2\})$

I soli insiemi sono quello vuoto o quelli ottenuti aggiungendo qualcosa ad un insieme

$\neg\exists\, x, s \;\; \{x|s\} = \{\}$ — l'insieme vuoto non si può decomporre

Assiomatizzazione del concetto di Insieme

Simbolo costante: $\{\}$ — insieme vuoto
Predicato unario: $Set(s)$ — insieme s
Predicato binario: $x \in s$ — scrittura infissa per $\in (x, s)$, x è un membro di s
Predicato binario: $s_1 \subseteq s_2$ — s_1 è un sottoinsieme (non necess. proprio) di s_2
Funzione binaria: $s_1 \cup s_2$ — unione di s_1 e s_2
Funzione binaria: $s_1 \cap s_2$ — intersezione di s_1 e s_2
Funzione binaria: $\{x|s_2\}$ — insieme risultante dall'aggiungere x all'insieme s

$\forall\, s \;\; Set(s) \;\Leftrightarrow\; (s = \{\}) \vee (\exists\, x, s_2 \;\; Set(s_2) \wedge s = \{x|s_2\})$

I soli insiemi sono quello vuoto o quelli ottenuti aggiungendo qualcosa ad un insieme

$\neg\exists\, x, s \;\; \{x|s\} = \{\}$ — l'insieme vuoto non si può decomporre

$\forall\, x, s \;\; x \in s \;\Leftrightarrow\; s = \{x|s\}$ — aggiungere un elemento già contenuto non ha effetto

Assiomatizzazione del concetto di Insieme

Simbolo costante: $\{\}$ — insieme vuoto
Predicato unario: $Set(s)$ — insieme s
Predicato binario: $x \in s$ — scrittura infissa per $\in (x, s)$, x è un membro di s
Predicato binario: $s_1 \subseteq s_2$ — s_1 è un sottoinsieme (non necess. proprio) di s_2
Funzione binaria: $s_1 \cup s_2$ — unione di s_1 e s_2
Funzione binaria: $s_1 \cap s_2$ — intersezione di s_1 e s_2
Funzione binaria: $\{x|s_2\}$ — insieme risultante dall'aggiungere x all'insieme s

$\forall s \;\; Set(s) \;\Leftrightarrow\; (s = \{\}) \vee (\exists x, s_2 \;\; Set(s_2) \wedge s = \{x|s_2\})$
I soli insiemi sono quello vuoto o quelli ottenuti aggiungendo qualcosa ad un insieme
$\neg\exists x, s \;\; \{x|s\} = \{\}$ — l'insieme vuoto non si può decomporre
$\forall x, s \;\; x \in s \;\Leftrightarrow\; s = \{x|s\}$ — aggiungere un elemento già contenuto non ha effetto
$\forall x, s \;\; x \in s \;\Leftrightarrow\; (\exists y, s_2 \;\; (s = \{y|s_2\} \wedge (x = y \vee x \in s_2)))$
unici elementi di un insieme sono quelli che sono stati aggiunti

Assiomatizzazione del concetto di Insieme

Simbolo costante: $\{\}$ — insieme vuoto
Predicato unario: $Set(s)$ — insieme s
Predicato binario: $x \in s$ — scrittura infissa per $\in (x, s)$, x è un membro di s
Predicato binario: $s_1 \subseteq s_2$ — s_1 è un sottoinsieme (non necess. proprio) di s_2
Funzione binaria: $s_1 \cup s_2$ — unione di s_1 e s_2
Funzione binaria: $s_1 \cap s_2$ — intersezione di s_1 e s_2
Funzione binaria: $\{x|s_2\}$ — insieme risultante dall'aggiungere x all'insieme s

$\forall s \;\; Set(s) \;\Leftrightarrow\; (s = \{\}) \vee (\exists x, s_2 \;\; Set(s_2) \wedge s = \{x|s_2\})$

I soli insiemi sono quello vuoto o quelli ottenuti aggiungendo qualcosa ad un insieme

$\neg\exists x, s \;\; \{x|s\} = \{\}$ — l'insieme vuoto non si può decomporre

$\forall x, s \;\; x \in s \;\Leftrightarrow\; s = \{x|s\}$ — aggiungere un elemento già contenuto non ha effetto

$\forall x, s \;\; x \in s \;\Leftrightarrow\; (\exists y, s_2 \;\; (s = \{y|s_2\} \wedge (x = y \vee x \in s_2)))$

unici elementi di un insieme sono quelli che sono stati aggiunti

$\forall s_1, s_2 \;\; s_1 \subseteq s_2 \;\Leftrightarrow\; (\forall x \;\; x \in s_1 \;\Rightarrow\; x \in s_2)$

Assiomatizzazione del concetto di Insieme

Simbolo costante: $\{\}$ — insieme vuoto
Predicato unario: $Set(s)$ — insieme s
Predicato binario: $x \in s$ — scrittura infissa per $\in (x, s)$, x è un membro di s
Predicato binario: $s_1 \subseteq s_2$ — s_1 è un sottoinsieme (non necess. proprio) di s_2
Funzione binaria: $s_1 \cup s_2$ — unione di s_1 e s_2
Funzione binaria: $s_1 \cap s_2$ — intersezione di s_1 e s_2
Funzione binaria: $\{x|s_2\}$ — insieme risultante dall'aggiungere x all'insieme s

$\forall s \;\; Set(s) \;\Leftrightarrow\; (s = \{\}) \vee (\exists x, s_2 \;\; Set(s_2) \wedge s = \{x|s_2\})$
I soli insiemi sono quello vuoto o quelli ottenuti aggiungendo qualcosa ad un insieme
$\neg\exists x, s \;\; \{x|s\} = \{\}$ — l'insieme vuoto non si può decomporre
$\forall x, s \;\; x \in s \;\Leftrightarrow\; s = \{x|s\}$ — aggiungere un elemento già contenuto non ha effetto
$\forall x, s \;\; x \in s \;\Leftrightarrow\; (\exists y, s_2 \;\; (s = \{y|s_2\} \wedge (x = y \vee x \in s_2)))$
unici elementi di un insieme sono quelli che sono stati aggiunti
$\forall s_1, s_2 \;\; s_1 \subseteq s_2 \;\Leftrightarrow\; (\forall x \;\; x \in s_1 \;\Rightarrow\; x \in s_2)$
$\forall s_1, s_2 \;\; (s_1 = s_2) \;\Leftrightarrow\; (s_1 \subseteq s_2 \wedge s_2 \subseteq s_1)$

Assiomatizzazione del concetto di Insieme

Simbolo costante: $\{\}$ — insieme vuoto
Predicato unario: $Set(s)$ — insieme s
Predicato binario: $x \in s$ — scrittura infissa per $\in (x, s)$, x è un membro di s
Predicato binario: $s_1 \subseteq s_2$ — s_1 è un sottoinsieme (non necess. proprio) di s_2
Funzione binaria: $s_1 \cup s_2$ — unione di s_1 e s_2
Funzione binaria: $s_1 \cap s_2$ — intersezione di s_1 e s_2
Funzione binaria: $\{x|s_2\}$ — insieme risultante dall'aggiungere x all'insieme s

$\forall s \;\; Set(s) \;\Leftrightarrow\; (s = \{\}) \vee (\exists x, s_2 \;\; Set(s_2) \wedge s = \{x|s_2\})$

I soli insiemi sono quello vuoto o quelli ottenuti aggiungendo qualcosa ad un insieme

$\neg\exists x, s \;\; \{x|s\} = \{\}$ — l'insieme vuoto non si può decomporre

$\forall x, s \;\; x \in s \;\Leftrightarrow\; s = \{x|s\}$ — aggiungere un elemento già contenuto non ha effetto

$\forall x, s \;\; x \in s \;\Leftrightarrow\; (\exists y, s_2 \;\; (s = \{y|s_2\} \wedge (x = y \vee x \in s_2)))$

unici elementi di un insieme sono quelli che sono stati aggiunti

$\forall s_1, s_2 \;\; s_1 \subseteq s_2 \;\Leftrightarrow\; (\forall x \;\; x \in s_1 \;\Rightarrow\; x \in s_2)$

$\forall s_1, s_2 \;\; (s_1 = s_2) \;\Leftrightarrow\; (s_1 \subseteq s_2 \wedge s_2 \subseteq s_1)$

$\forall x, s_1, s_2 \;\; x \in (s_1 \cap s_2) \;\Leftrightarrow\; (x \in s_1 \wedge x \in s_2)$

Assiomatizzazione del concetto di Insieme

Simbolo costante: $\{\}$ — insieme vuoto
Predicato unario: $Set(s)$ — insieme s
Predicato binario: $x \in s$ — scrittura infissa per $\in (x, s)$, x è un membro di s
Predicato binario: $s_1 \subseteq s_2$ — s_1 è un sottoinsieme (non necess. proprio) di s_2
Funzione binaria: $s_1 \cup s_2$ — unione di s_1 e s_2
Funzione binaria: $s_1 \cap s_2$ — intersezione di s_1 e s_2
Funzione binaria: $\{x|s_2\}$ — insieme risultante dall'aggiungere x all'insieme s

$\forall s \;\; Set(s) \;\Leftrightarrow\; (s = \{\}) \vee (\exists x, s_2 \;\; Set(s_2) \wedge s = \{x|s_2\})$

I soli insiemi sono quello vuoto o quelli ottenuti aggiungendo qualcosa ad un insieme

$\neg\exists x, s \;\; \{x|s\} = \{\}$ — l'insieme vuoto non si può decomporre

$\forall x, s \;\; x \in s \;\Leftrightarrow\; s = \{x|s\}$ — aggiungere un elemento già contenuto non ha effetto

$\forall x, s \;\; x \in s \;\Leftrightarrow\; (\exists y, s_2 \;\; (s = \{y|s_2\} \wedge (x = y \vee x \in s_2)))$

unici elementi di un insieme sono quelli che sono stati aggiunti

$\forall s_1, s_2 \;\; s_1 \subseteq s_2 \;\Leftrightarrow\; (\forall x \;\; x \in s_1 \;\Rightarrow\; x \in s_2)$

$\forall s_1, s_2 \;\; (s_1 = s_2) \;\Leftrightarrow\; (s_1 \subseteq s_2 \wedge s_2 \subseteq s_1)$

$\forall x, s_1, s_2 \;\; x \in (s_1 \cap s_2) \;\Leftrightarrow\; (x \in s_1 \wedge x \in s_2)$

$\forall x, s_1, s_2 \;\; x \in (s_1 \cup s_2) \;\Leftrightarrow\; (x \in s_1 \vee x \in s_2)$

Interazione con KB in FOL

Supponiamo che l'agente operante nel modo dei Wumpus utilizzi una
KB in FOL e che questi percepisca puzza e brezza (ma non luccichio) all'istante di tempo $t = 5$ (rappresentiamo gli istanti di tempo attraverso i numeri interi):

$Tell(KB, Percepito([Puzza, Brezza, Niente], 5))$
$Ask(KB, \exists\, a \;\; Azione(a, 5))$

Cioè, la KB ha come conseguenza logica qualche azione particolare a tempo $t = 5$?

Risposta: $Si,\ \{a/Shoot\}$ $\leftarrow$ sostituzione (lista di assegnamenti)

Data una sentenza S ed una sostituzione σ,
$S\sigma$ denota il risultato dell'inserimento di σ in S; p.e.,
$S = PiuIntelligente(x, y)$
$\sigma = \{x/Uomo, y/Scimmia\}$
$S\sigma = PiuIntelligente(Uomo, Scimmia)$

$Ask(KB, S)$ restituisce alcune/tutte le σ tali che $KB \models S\sigma$

Base di Conoscenza per il Mondo dei Wumpus

"Percezione"

$\forall\, b, g, t \;\; Percepito([Puzza, b, g], t) \;\Rightarrow\; AnnusataPuzza(t)$

$\forall\, s, b, t \;\; Percepito([s, b, Luccichio], t) \;\Rightarrow\; RaggiuntoOro(t)$

Riflesso: $\forall\, t \;\; RaggiuntoOro(t) \;\Rightarrow\; Azione(Prendi, t)$

Riflesso con Stato Interno: Oro già preso ?

$\forall\, t \;\; RaggiuntoOro(t) \wedge \neg Mantenuto(Oro, t) \;\Rightarrow\; Azione(Prendi, t)$

$Mantenuto(Oro, t)$ non può essere osservato

$\Rightarrow$ mantenere traccia dei cambiamenti è essenziale

Deduzione di Proprietà Nascoste

Proprietà di locazioni:
$\forall x,t \;\; Luogo(Agente,x,t) \wedge AnnusataPuzza(t) \;\Rightarrow\; Puzzolente(x)$
$\forall x,t \;\; Luogo(Agente,x,t) \wedge Brezza(t) \;\Rightarrow\; Ventilato(x)$

I quadrati sono ventilati in prossimità di una trappola:

Regola Diagnostica —inferisce la causa dagli effetti
$$\forall y \;\; Ventilato(y) \;\Rightarrow\; \exists x \;\; Trappola(x) \wedge Adiacente(x,y)$$

Regola Causale —inferisce l'effetto dalla causa
$$\forall x,y \;\; Trappola(x) \wedge Adiacente(x,y) \;\Rightarrow\; Ventilato(y)$$

Nessuna di queste regole è completa—p.e., la regola causale non dice se i quadrati lontani dalla trappola sono ventilati

Definizione per il predicato $Ventilato$:
$$\forall y \;\; Ventilato(y) \;\Leftrightarrow\; [\exists x \;\; Trappola(x) \wedge Adiacente(x,y)]$$

Mantenere traccia dei cambiamenti

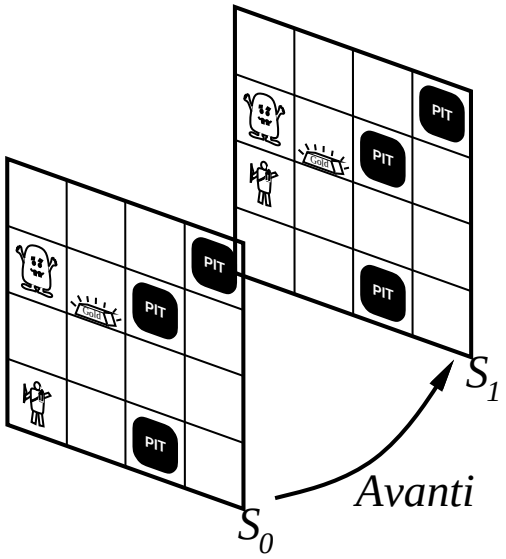
I fatti sono veri in situazioni, piuttosto che eternamente
E.g., $Mantenuto(Oro, Adesso)$ piuttosto che $Mantenuto(Oro)$

Calcolo delle Situazioni è un modo di rappresentare in FOL i cambiamenti:

Aggiunge un argomento di situazione a ogni predicato non eterno
P.e., $Adesso$ in $Mantenuto(Oro, Adesso)$ denota una situazione

Le situazioni sono connesse tramite la funzione $Risultato$

$Risultato(a, s)$ è la situazione che risulta nell'eseguire l'azione a nella situazione s



Descrizione delle Azioni I

Assioma "Effetto" —descrive i cambiamenti dovuti all'azione

$\forall\, s \;\; RaggiuntoOro(s) \;\Rightarrow\; Mantenuto(Oro, Risultato(Prendi, s))$

Assioma "Frame" —descrive non-cambiamenti dovuti all'azione

$\forall\, s \;\; HaFreccia(s) \;\Rightarrow\; HaFreccia(Risultato(Prendi, s))$

Frame problem: trovare un modo elegante per gestire il non-cambiamento

(a) rappresentazione—evitare assiomi di frame:

$O(\#pred \times \#az)$, invece di $O(\#pred \times \#eff)$

(b) inferenza—evitare copie ripetute per mantenere traccia dello stato:

dati t passi, $O(\#pred \times t)$ o $O(\#pred \times \#eff \times t)$, invece di $O(\#eff \times t)$

Problema della Qualifica: descrizioni fedeli di azioni reali richiedono dettagli senza fine—cosa succede se l'oro è scivoloso o inchiodato a terra o . . .

Problema della Ramificazione: le azioni reali hanno molte conseguenze secondarie—spostamento agente con oro implica spostamento oro, . . .

Descrizione delle Azioni II

assiomi di stato-successore risolvono il problema del frame sulla rappresentazione

Ogni assioma "riguarda" un predicato (non una azione per se):

P vero successivamente $\Leftrightarrow$ [una azione rende P vero
$\vee$ P già vero e nessuna azione rende P falso]

Mantenimento dell'oro:

$$\forall a, s \;\; Mantenuto(Oro, Risultato(a, s)) \Leftrightarrow$$
$$[(a = Prendi \wedge RaggiuntoOro(s))$$
$$\vee (Mantenuto(Oro, s) \wedge a \neq Lascia)]$$

Pianificazione

Condizioni iniziali in KB:

$Luogo(Agente, [1,1], S_0)$
$Luogo(Oro, [1,2], S_0)$

Query: $Ask(KB, \exists\, s \;\; Mantenuto(Oro, s))$

cioè, in quale situazione manterrò l'oro ?

Risposta: $\{s/Risultato(Prendi, Risultato(Avanti, S_0))\}$

cioè, vai avanti e poi prendi l'oro

Si assume che l'agente sia interessato ad iniziare una pianificazione nella situazione S_0 e che S_0 sia l'unica situazione descritta nella KB

Pianificazione: una soluzione migliore

Representare piani come sequenze di azioni $[a_1, a_2, \ldots, a_n]$

$RisultatoPiano(p, s)$ è il risultato che si ottiene eseguendo p in s

Allora la query $Ask(KB, \exists p \;\; Mantenuto(Oro, RisultatoPiano(p, S_0)))$
ha soluzione $\{p/[Avanti, Prendi]\}$

Definizione di $RisultatoPiano$ in termini di $Risultato$:

$\forall s \;\; RisultatoPiano([\,], s) = s$

$\forall a, p, s \;\; RisultatoPiano([a|p], s) = RisultatoPiano(p, Risultato(a, s))$

Planning systems: sono sistemi special-purpose progettati per fare questo tipo di inferenza più efficientemente di sistemi general-purpose

Riassunto

Logica del Primo Ordine:

– oggetti e relazioni sono primitive semantiche

– sintassi: costanti, variabili, funzioni, predicati, uguaglianza, quantificatori

Aumentato potere espressivo: sufficiente per definire il mondo dei wumpus

Calcolo delle situazioni:

– convenzioni per descrivere azioni e cambiamenti in FOL

– permette di generare piani tramite inferenza su una "situation calculus" KB