Intelligenza Artificiale e Laboratorio

Programmazione logica e Ragionamento non monotono

Alberto Martelli

Programmazione logica

La negazione in Prolog non è la negazione classica, ma negazione per falllimento - **negation as failure (NAF).**

Un goal negato **?- not p** è valutato secondo la seguente regola **(NAF)**:

?- not p ha successo se **?- p** fallisce finitamente;

?- not p fallisce se **?- p** ha successo.

La negazione per fallimento introduce la non monotonicità nel Logic Programming: un goal può avere successo da un programma P, ma fallire se si aggiungono altre clausole a P.

Negation as failure

fly(X) ← bird(X) ∧ **not** abnormal(X)
bird(X) ← penguin(X)
bird(tweety)
abnormal(X) ← penguin(X)

Il goal ?- fly(tweety) ha successo perché il goal

?- abnormal(tweety)

fallisce finitamente.

Se aggiungiamo il fatto penguin(tweety), il goal ?- fly(tweety) fallisce perché il goal

?- abnormal(tweety)

ha successo.

Quale semantica per NAF

La negazione per fallimento ha una **semantica operazionale** (data dall' interprete)

E' possibile dare una **semantica dichiarativa** alla programmazione logica con negazione?

Ci sono strette relazioni fra alcuni approcci alla semantica della programmazione logica con negazione e le logiche non monotone (in particolare la logica di default).

Semantica di NAF: completamento [Clark 78]

Dato un **programma logico generale (con negazione) P**, il **completamento** di **P comp(P)** è un insieme di formule della logica classica tali che gli atomi derivabili da P sono conseguenze logiche (in logica classica) di comp(P).

Attraverso il completamento, le condizioni **sufficienti** per la verità di un atomo, rappresentate dalle clausole del programma, sono considerate anche come condizioni **necessarie**.

Il se ($\Leftarrow$) viene sostituito con il se e solo se ($\Leftrightarrow$).

Esempio completamento

P:

$vola(X) \Leftarrow uccello(X) \wedge$ **not** $abnormal(X)$

$uccello(X) \Leftarrow pinguino(X)$

$abnormal(X) \Leftarrow pinguino(X)$

$uccello(tweety)$

comp(P):

$vola(X) \Leftrightarrow uccello(X) \wedge \neg abnormal(X)$

$uccello(X) \Leftrightarrow pinguino(X) \vee X=tweety$

$abnormal(X) \Leftrightarrow pinguino(X)$

$\neg pinguino(X)$

vola(tweety) è derivabile da P con la negazione per fallimento, ed è derivabile classicamente da comp(P) perché ¬abnormal(tweety) è derivabile.

Se si aggiunge a P pinguino(tweety), la formula relativa a pinguino nel completamento diventa $pinguino(X) \Leftrightarrow X=tweety$.

Completamento

Nel caso proposizionale il completamento comp(P) di un programma P per ogni proposizione q definita in P dalle clausole

$$q \Leftarrow E_1$$
$$\ldots\ldots..$$
$$q \Leftarrow E_n$$

contiene la formula

$$q \Leftrightarrow E_1 \vee \ldots.. \vee E_n$$

Se q non è definita in P, allora comp(P) contiene $\neg q$.
Nel caso predicativo, comp(P) deve anche contenere assiomi sull'uguaglianza per l'ipotesi dei nomi unici (due costanti sono uguali solo se hanno lo stesso nome).

Per i programmi positivi, la derivazione da P è corretta e completa rispetto alla derivazione da comp(P).

Problemi con il completamento

Se P è un programma generale, comp(P) può essere inconsistente.

$P=\{q \Leftarrow \textbf{not } q\}$ $\quad\quad$ $comp(P)=\{q \Leftrightarrow \neg q\}$

comp(P) è inconsistente.

Dato il programma

$$P=\{ \quad p \Leftarrow q$$
$$p \Leftarrow \textbf{not } q$$
$$q \Leftarrow q\}$$

la derivazione di p in Prolog va in loop, mentre comp(P) $\models$ p.

E' possibile definire restrizioni sintattiche sui programmi per garantire consistenza e completezza (es. stratificazione).
E' stato anche proposto di passare dalla logica a due valori a quella a tre.

Semantica di NAF: Modelli stabili

[Gelfond & Lifschitz 88]

Un **modello stabile** è un modello **minimo** in cui ogni atomo ha una giustificazione (supporto), ovvero esiste una regola di cui l'atomo è la testa e ogni letterale del corpo è soddisfatto.

P: vola(tweety) ⇐ uccello(tweety) ∧ **not** eccezione(tweety)
uccello(tweety) ⇐ pinguino(tweety)
eccezione(tweety) ⇐ pinguino(tweety)
uccello(tweety)

S = {uccello(tweety), vola(tweety)} è un modello stabile di P.

Programmi logici positivi

Cominciamo a considerare

Programmi logici positivi (senza negazione) proposizionali.

E' possibile definire una **semantica model-theoretic** basata sulla relazione di conseguenza logica ($\models$) della logica classica.

Ricordiamo che un **modello** di una formula α è un insieme di simboli proposizionali in cui α è vera.

Ad esempio la formula $\alpha = (a \lor b)$ ha i modelli $\{a\}$, $\{b\}$ e $\{a,b\}$.

Programmi logici positivi

La semantica model-theoretic di un programma positivo P è l'insieme di simboli proposizionali Decl(P) tale che

$$\mathrm{Decl}(P) = \{A \mid P \models A\}$$

Decl(P) è un modello.

Per i programmi logici positivi si può dimostrare che l'intersezione di due modelli è a sua volta un modello, e quindi esiste un **modello minimo** M(P).

Decl(P) è proprio questo modello minimo: Decl(P) = M(P).

Esempio

Sia dato il programma:

P: $c \Leftarrow a \wedge b$

a

Il programma ha diversi modelli nella logica classica:

$\{a, b, c\}$ $\{a, c\}$ $\{a\}$

Il modello minimo è $\{a\}$.

a è l'unica conseguenza logica del programma P.

Semantica di NAF: modelli stabili

Consideriamo adesso il caso generale.
Sia **P** un **programma logico generale (con negazione) proposizionale** e **S** un insieme di proposizioni.

Il **ridotto** $\mathbf{P_S}$ di **P** rispetto ad **S** è il programma ottenuto da **P** cancellando:

(a) tutte le regole il cui corpo contiene un letterale **not** q, con $q \in \mathbf{S}$
(b) tutti i letterali negativi nelle regole rimanenti.

$\mathbf{P_S}$ è un programma positivo e quindi ha un modello minimo $M(\mathbf{P_S})$.

S è un **modello stabile** se $\mathbf{S} = M(\mathbf{P_S})$.

S è un **punto fisso** dell'equazione $\mathbf{S} = M(\mathbf{P_S})$.

Modelli stabili: esempi

P: $\quad$ vola(tweety) $\Leftarrow$ uccello(tweety) $\wedge$ **not** eccezione(tweety)
uccello(tweety) $\Leftarrow$ pinguino(tweety)
eccezione(tweety) $\Leftarrow$ pinguino(tweety)
uccello(tweety)

S = {uccello(tweety), vola(tweety)}è un modello stabile di P.
Infatti:
P_S: $\quad$ vola(tweety) $\Leftarrow$ uccello(tweety)
uccello(tweety) $\Leftarrow$ pinguino(tweety)
eccezione(tweety) $\Leftarrow$ pinguino(tweety)
uccello(tweety)

$M(P_S)$={uccello(tweety), vola(tweety)}=S

S è l'unico modello stabile di P.

Modelli stabili: esempi

P: $p \Leftarrow r \wedge$ **not** q

$p \Leftarrow s$

$q \Leftarrow$ **not** s

$r \Leftarrow r$

$S=\{q\}$ è un modello stabile di P:

P_S: $p \Leftarrow s$

q

$r \Leftarrow r$

$M(P_S)=S$

$P=\{p \Leftarrow$ **not** $p\}$non ha modelli stabili

$P=\{p \Leftarrow$ **not** q, $q \Leftarrow$ **not** $p\}$ ha due modelli stabili $S1=\{p\}$ e $S2=\{q\}$

Proprietà dei modelli stabili

Per ogni modello stabile **S** di un programma logico **P**:
se $l \in \mathbf{S}$ allora l è supportato da **P**, ossia esiste una regola di **P**

$$l \Leftarrow l_1 \wedge \ldots \wedge l_m \wedge \textbf{not } l_{m+1} \wedge \ldots \wedge \textbf{not } l_n$$

tale che $\{l_1, \ldots, l_m\} \subseteq S$ e $\{l_{m+1}, \ldots, l_n\} \cap S = \varnothing$

Per ogni programma **P**, se $\mathbf{S_0}$ e $\mathbf{S_1}$ sono modelli stabili di **P**, non può essere $\mathbf{S_0} \subset \mathbf{S_1}$ (i modelli stabili sono minimi).

Modelli stabili

La semantica con modelli stabili è definita per un programma P, se P ha un unico modello stabile. Questo modello può essere considerato come il modello canonico di P.

E se P non ha modelli o ne ha più di uno?

Si potrebbero adottare le modalità di conseguenza logica della logica di default *scettica* o *credula*, facendo riferimento rispettivamente a tutti i modelli o a uno solo.

Modelli stabili e logica di default

La regola

$$\text{mario\_lavora} \Leftarrow \text{giorno\_feriale} \wedge \textbf{not}\ \text{mario\_malato}$$

può essere interpretata in logica di default come

$$\frac{\text{giorno\_feriale} :\ \neg \text{mario\_malato}}{\text{mario\_lavora}}$$

In generale, è possibile associare a un programma logico P una teoria di default dove:

- ogni clausola è sostituita con una regola di default
- per ogni atomo A in P, si introduce una regola di default di mondo chiuso:

$$\frac{:\ \neg A}{\neg A}$$

S è un modello stabile di P sse S è un modello di una estensione della teoria di default associata a P.