$(s,[A],t) \in R^c$ sse $\forall B$ $(A>B \in s \Rightarrow B \in t)$, avendo posto $[A]=\{s \in S^c: A \in s\}$. Si dimostra che R^c soddisfa le proprietà (24'), (25') e (26'). Chiaramente, $\mathcal{S}^c$ non è una γ_0–struttura perchè una terna $(s, Q, t) \in R$ solo se Q ha la forma [A] e questo impedisce la soddisfazione delle condizioni (25') e (26'). Tuttavia, usando la tecnica delle filtrazioni, si può dimostrare che per ogni A che non sia un teorema di γ_0 esiste una struttura finita $|\mathcal{S}|=(|S|, |R|)$, filtrata su $\mathcal{S}^c$, nella quale A viene falsificata. I dettagli della dimostrazione saranno, come accennato in precedenza, oggetto di un lavoro successivo.

***

BIBLIOGRAFIA

[1] FISCHER SERVI, G. *Panoramica sulle logiche intuizioniste modali*, Quaderni di Filosofia, 11 Parma (1987).

[2] GABBAY, D. *Intuitionistic basis for Non Monotonic Logic*, in *Proc. Conf. on automated deduction*, Springer Lecture Notes in Computer Science, 6 (1982).

[3] MANGIONE, C. *Elementi di logica matematica*, Boringhieri 1964.

[4] Mc DERMOTT, D. and DOYLE, J. *Non Monotonic Logic* 1, Artificial Intelligence, 13, 1980.

[5] Mc DERMOTT, D. Non Monotonic Logic 2: *Non Monotonic Theories*, JACM, 29, 1, 1982.

[6] REITER, R. *A logic for default reasoning*, Artificial Intelligence, 13, 1980.

STRUTTURE DI SIMULAZIONE COME AMBIENTE SEMANTICO PER UN CALCOLO CLASSICO

Alessandra Carbone[1]

Dipartimento di Scienze dell'Informazione, Università di Milano

SOMMARIO

La necessità di trattare la mancanza di conoscenza (*sintattico-semantica*) nella deduzione e l'esplicito legame presente nel sistema FOL (Weyhrauch, 1980) tra l'ambiente di rappresentazione sintattico e quello *semantico* (o meglio *computazionale*), hanno indotto alla formalizzazione - qui presentata - di una semantica associata a un calcolo dei predicati classico, capace di individuare la mancanza di interpretazione dei simboli di un linguaggio logico del primo ordine. A tal fine è stata usata la nozione di struttura di simulazione (già presente in FOL nelle vesti di *modello computazionale*) come nucleo della nuova semantica e sono state date le correlazioni tra quest'ultima e quella classica.

1.INTRODUZIONE

L'interazione tra conoscenza dichiarativa e conoscenza procedurale nei sistemi di intelligenza artificiale ha un'importante esempio in FOL. (First Order Logic). In questo sistema le due forme di conoscenza interagiscono per permettere la valutazione (sintattica e/o semantica) di una formula ben formata di un linguaggio logico del primo ordine.

Brevemente, FOL nasce come proof checker (Weyhrauch, 1977) che implementa un'estensione del calcolo classico della deduzione naturale descritto da Prawitz (1965). Oltre ad un ambiente di deduzione puramente sintattico, FOL fornisce uno strumento per la costruzione di un ambito in cui possa venir rappresentata conoscenza algoritmica: l'attachment semantico.
Tale strumento collega i simboli della segnatura di una teoria a strutture dati lisp e crea una *struttura di simulazione* o *modello parziale* di una teoria (così chiamato perchè non soddisfa

[1]Indirizzo attuale: Dipartimento di Matematica, Università di Siena.

le condizioni di chiusura sul modello come invece avviene classicamente).

Questo articolo analizza un ambiente logico inusuale quale risulta quello fornito dalle strutture di simulazione (Carbone, 1985). Al fine di individuare la mancanza di denotazione dei simboli del linguaggio, verrà determinata una semantica consistente con il calcolo classico, ma basata sul nuovo concetto semantico piuttosto che su quello di modello classico.

In letteratura si trovano le logiche libere e le logiche a più valori di verità che si preoccupano di trattare il problema della presenza di informazione incompleta, utilizzando, però, un calcolo non classico. Le prime sono state studiate da Van Fraassen (Van Fraassen,1966a; 1966b; 1968) e, sulla stessa linea di ricerca, da Bencivenga (Bencivenga, 1976). L'obiettivo da loro perseguito è stato quello di esprimere un formalismo per semantiche non bivalenti, cioè semantiche che interpretino l'espressione *ne' vero ne' falso* : le cosiddette semantiche con lacune. Dire che un enunciato è *ne' vero ne' falso* equivale a dire che è privo di valore di verità, non valutato.
Un'alternativa a questo modo di procedere è stata data da coloro che hanno interpretato l'enunciato *ne' vero ne' falso* con un altro valore di verità che naturalmente non è ne' 1 ne' 0 ma un terzo valore perfettamente definito. La caratterizzazione della semantica trivalente è dovuta a Lukasiewiecz (Lukasiewiecz, 1920; Lukasiewiecz, 1930) ed altri l'hanno estesa introducendo n valori di verità distinti da 1 e 0, per qualche n.
In tali logiche il calcolo è più restrittivo e quindi ci sono teoremi classici che, in questi ambiti, non possono essere dimostrati.

Poichè FOL è caratterizzato da un calcolo classico, è stato necessario trovare un'alternativa adatta affinchè fosse possibile cogliere pienamente le caratteristiche del sistema. Ad appoggiare questo primo fondamentale motivo, ha avuto rilevanza il fatto che l'uso delle logiche non classiche presentate, consentirebbe di trattare al loro interno, come *conseguenze logiche*, gli effetti derivanti dai fenomeni di non denotanza, mentre nel caso di FOL l'obiettivo era quello di conciliare la mancanza di informazione con l'ambito classico. Queste ragioni ci hanno quindi condotto ad individuare un nuovo approccio per trattare tali aspetti allo scopo di studiare se la nozione semantica di modello parziale potesse essere usata in modo consistente col calcolo, e di stabilire possibili vantaggi nell'uso dei due differenti tipi di conoscenza (procedurale e dichiarativa) nel sistema.

Una semantica formalizzata per questi concetti:

- permetterà un'analisi e una valutazione precisa dei meccanismi di semplificazione semantico/sintattica (Aiello, 1980);
- apre possibili aree di ricerca sulla costruzione automatica di strutture di simulazione (Carbone, 1985).

La semantica proposta si estende all'intero calcolo dei predicati, mentre le valutazioni realizzabili in FOL attraverso l'impiego di strutture di simulazione sono ristrette alle formule proposizionali.

2. ANALISI DELLE STRUTTURE DI SIMULAZIONE NELLA LOGICA CLASSICA

In questa sezione chiariremo la nozione di struttura di simulazione usata in FOL e proporremo una nozione di validità estesa alle formule ben formate (fbf) quantificate. Questa estensione non è considerata in FOL, ma risulta interessante da un punto di vista teorico. La nozione di validità permetterà di giustificare poi, il calcolo classico presente in FOL; per una discussione più dettagliata su tale argomento rimandiamo alle conclusioni mentre proponiamo ora le definizioni e le principali proprietà che seguono da tale nozione.

2.1 CLASSI DI STRUTTURE DI SIMULAZIONE E GIUSTIFICAZIONE SEMANTICA DELL'USO DELLA NOZIONE CLASSICA DI VALIDITA'

Diamo per nota la nozione di linguaggio logico del primo ordine L (Chang, Keisler, 1973), e adottiamo il seguente formalismo:

$$L = \langle F(L), P(L), C(L), V(L) \rangle$$

dove F(L), P(L), C(L), V(L) rappresentano l'insieme dei simboli di funzione, di predicato, di costante e di variabile nel linguaggio L, rispettivamente.

Diamo per nota anche la nozione di modello classico e procediamo con l'estendere tale definizione al caso delle strutture parziali quali sono le strutture di simulazione in FOL.

Def 2.1: Fissato il linguaggio L, definiamo una struttura per L, essere un attachment semantico SS sull'insieme D_{SS} (chiamato *supporto* o *dominio della struttura*) che soddisferà le seguenti proprietà:

1) per ogni $c \in C_{SS}$ con $C(L) \supseteq C_{SS}$ (C_{SS} è un sottinsieme delle costanti del linguaggio) è definito un attachment:

$$SS(c) \in D_{SS};$$

C_{SS} è l'insieme dei simboli di costante denotati;

2) per ogni $f \in F_{SS}$ con $F(L) \supseteq F_{SS}$ (F_{SS} è un sottinsieme dei simboli di funzione) è definito un attachment:

Come ulteriore esempio di algoritmo di rilassamento, consideriamo l'algoritmo di *path-consistency* definito da Montanari in [9]. esso si applica a reti di vincoli binarie e complete, quindi G è un grafo completo contenente tutti i vincoli unari a_i, con $c(a_i) = \langle x_i \rangle$, più tutti i vincoli binari a_{ij}, con $c(a_{ij}) = \langle x_i,x_j \rangle$. $R = \{a_{ij}(x_i,x_j) \leftarrow a_{ik}(x_i,x_k), a_k(x_k), a_{kj}(x_k,x_j)$, per $i,j,k=1,\ldots,\#N\}$ e, di nuovo, in generale non è adeguato per C. Una generica regola di rilassamento dell'algoritmo di *path-consistency* può essere vista in Fig.3.

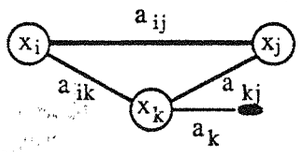

Fig.3. Una regola di rilassamento per l'algoritmo di *path-consistency*.

Come esempio di rete di vincoli in cui le regole di *arc-consistency* non sono adeguate, consideriamo la rete in Fig.4 dove tutti gli archi sono etichettati dalla relazione "≠" e U = {0,1}. Tale rete rappresenta il ben conosciuto problema di colorare un grafo completo di tre nodi con due colori. Come sappiamo, questo problema non ha soluzione, quindi la soluzione della rete è la relazione vuota. Invece, la rete è già stabile rispetto alle regole di *arc-consistency*, in quanto, per ogni valore di un nodo x, esiste un valore per un qualsiasi altro nodo y tale che il vincolo tra x e y è soddisfatto. Al contrario, le regole di *path-consistency* sono adeguate, cioè sufficienti per risolvere la rete.

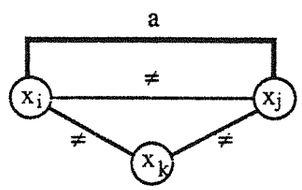

Fig.4. Le regole di *arc-consistency* possono non essere adeguate.

Vediamo ora un esempio, in Fig.5, in cui ne' le regole di *arc-consistency* ne' quelle di *path-consistency* sono adeguate. Qui U = {1,2,3} e tutti gli archi sono etichettati dalla relazione "≠". Tale rete rappresenta il ben conosciuto problema di colorare un grafo completo di quattro nodi con tre colori.

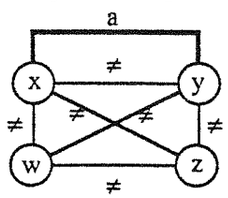

Fig.5. Le regole di *arc-consistency* e di *path-consistency* possono essere non adeguate.

b) 1<2 per l'attachment di <, 1, 2.

c) $0<1 \wedge 1<2 \rightarrow 0<2$ per istanziazione sull'assioma.

d) 0<2 per introduzione di $\wedge$ tra a) e b), e modus ponens con c).

A partire dall'insieme dei simboli di costante, funzione, predicato denotati, definiremo il concetto di *denotanza* per una fbf A su di una struttura di simulazione. Definiamo, innanzitutto, il significato di *termine denotato t* (chiameremo tale nozione *den t*). Definiremo poi il significato di *formuladenotata A* (chiameremo tale nozione *den A*). Osserviamo che il termine t e la formula A sono denotati *sulla* struttura di simulazione SS quindi sarebbe più corretto chiamarli den(t,SS) e den(A,SS) rispettivamente (questa osservazione vale anche per le nozioni di termine e formula definiti, introdotti in seguito).

Def 2.2:

- se $t = x$ e $x \in V(L)$ allora den t;
- se $t = c$ e $c \in C_{SS}$ allora den t;
- se $t = f^n(t_1,\ldots, t_n)$ allora den t sse $f^n \in F_{SS}$ e den t_i per $i = 1, \ldots, n$;
- nient'altro definisce la denotazione di un termine t.

Def 2.3:

base) se $A = p^n(t_1,\ldots, t_n)$ allora den A sse $p^n \in P_{SS}$ e den t_i per $i = 1, \ldots, n$;

passo) se $A = B \wedge C$ allora den A sse den B e den C

se $A = B \vee C$ allora den A sse den B e den C

se $A = \neg B$ allora den A sse den B

se $A = B \rightarrow C$ allora den A sse den B e den C

se $A = \forall x\, B$ allora den A sse den B

se $A = \exists x\, B$ allora den A sse den B

Stabiliamo a questo punto cosa si intende per formula definita rispetto ad una struttura di simulazione SS per un assegnamento $\underline{x} = \underline{\alpha}$. Precisiamo che con assegnamento $\underline{x} = \underline{\alpha}$ si intende una n-upla della forma:

$$\{ (x_1, \alpha_1), \ldots, (x_n, \alpha_n)\}$$

dove ogni $\alpha_i \in D_{SS}$ e α_i è detto il valore assegnato a x_i per $i = 1, \ldots, n$.
Definiamo cosa si intende con valore di un termine t (chiamato *val(t)*) per un assegnamento $\underline{x} = \underline{\alpha}$.

Def 2.4: per induzione sulla struttura di t,

base) $val(c, \underline{x} = \underline{\alpha}) = SS(c)$ se den c
$= \text{indefinito}$ altrimenti

$val(x_i, \underline{x} = \underline{\alpha}) = \alpha_i$ se $(x_i, \alpha_i) \in \underline{x} = \underline{\alpha}$
$= \text{indefinito}$ altrimenti

passo) $val(f(t_1,..., t_n), \underline{x} = \underline{\alpha}) = f'(\alpha_1, ..., \alpha_n)$ se per ogni $(1 \le i \le n)$ $\alpha_i = val(t_i, \underline{x} = \underline{\alpha})$ è definito e $f' = SS(f)$ e $(\alpha_1, ..., \alpha_n) \in dom(f')$[2]

$= \text{indefinito}$ altrimenti

Definiamo ora cosa si intende per valore di una fbf atomica $p(t_1,..., t_n)$ per un assegnamento $\underline{x} = \underline{\alpha}$.

Def 2.5: $val(p(t_1,..., t_n), \underline{x} = \underline{\alpha}) = p'(\alpha_1, ..., \alpha_n)$ se per ogni $(1 \le i \le n)$ $\alpha_i = val(t_i, \underline{x} = \underline{\alpha})$ è definito e $p' = SS(f)$ e $(\alpha_1, ..., \alpha_n) \in dom(p')$

$= \text{indefinito}$ altrimenti

Una fbf A è definita su SS per $\underline{x} = \underline{\alpha}$ se vale ciò che segue:

Def. 2.6

base) $A = p^n(t_1,..., t_n)$ e $val(p^n(t_1,..., t_n), \underline{x} = \underline{\alpha})$ è definito;

passo) $A = B \wedge C$ e B, C sono definite per $\underline{x} = \underline{\alpha}$

$A = B \vee C$ e B, C sono definite per $\underline{x} = \underline{\alpha}$

se $A = \neg B$ e B è definita per $\underline{x} = \underline{\alpha}$

se $A = B \rightarrow C$ e B, C sono definite per $\underline{x} = \underline{\alpha}$

se $A = \forall y\, B$ e B è definita per $\underline{x} = \underline{\alpha}$ e per ogni $y / \underline{x} = \underline{\alpha} = \beta$

se $A = \exists y\, B$ e B è definita per $\underline{x} = \underline{\alpha}$ e per ogni $y / \underline{x} = \underline{\alpha} = \beta$

Precisiamo che con il simbolo $y / \underline{x} = \underline{\alpha}$ si sottintende che la variabile y non compaia nella n-upla di variabili $\underline{x} = \underline{\alpha}$ (cioè $y \neq x_i$ per ogni $i = 1, ..., n$). Nel caso in cui $y = x_i$ per qualche i, si suppone che y venga rinominata. Con il simbolo $y / \underline{x} = \underline{\alpha} = \beta$ si intende dunque l'assegnamento a y del valore β. Tale osservazione dovrà venir conservata per ogni altra occorrenza di tale simbologia.

---

[2]Qui dom(f') rappresenta il dominio di terminazione di una funzione f'.

Data una fbf A definiamo $SS \models^{v} A$ per $\underline{x} = \underline{\alpha}$ come segue:

Def 2.7:

base) $A = p^n(t_1,..., t_n)$: $SS \models^{v} A$ sse $val(p^n(t_1,..., t_n), \underline{x} = \underline{\alpha}) = V$

passo) $A = B \wedge C$: $SS \models^{v} A$ sse $SS \models^{v} B$ e $SS \models^{v} C$ per $\underline{x} = \underline{\alpha}$

$A = B \vee C$: $SS \models^{v} A$ sse $SS \models^{v} B$ o $SS \models^{v} C$ per $\underline{x} = \underline{\alpha}$

se $A = \neg B$: $SS \models^{v} A$ sse non $SS \models^{v} B$ e B è definito su SS

se $A = B \rightarrow C$: $SS \models^{v} A$ sse non $SS \models B$ o $SS \models^{v} C$ per $\underline{x} = \underline{\alpha}$

se $A = \exists y\, B$: $SS \models^{v} A$ sse esiste $\beta \in D_{SS}$ tale che $SS \models^{v} B$ per $\underline{x} = \underline{\alpha}$ e $y / \underline{x} = \underline{\alpha} = \beta$

se $A = \forall y\, B$: $SS \models^{v} A$ sse per ogni $\beta \in D_{SS}$.$SS \models^{v} B$ per $\underline{x} = \underline{\alpha}$ e $y / \underline{x} = \underline{\alpha} = \beta$

Oss 2.2: in Def 2.7 possiamo osservare che la validità di $SS \models^{v} \neg A$ implica che la fbf A sia definita; cosa che non vale necessariamente per le fbf non negative (vedi Oss 2.3).

Oss 2.3: se A è indefinita, allora può valere sia non $SS \models^{v} A$ che non $SS \models^{v} \neg A$; tuttavia $SS \models^{v} A$ per $\underline{x} = \underline{\alpha}$ non implica necessariamente che A sia definita per $\underline{x} = \underline{\alpha}$; ad esempio, se A è definita per $\underline{x} = \underline{\alpha}$ e $SS \models^{v} A$, allora $SS \models^{v} A \vee B$ per ogni B, definita o meno. Ciò non vale in FOL, dove il valutatore semantico (chiamato SYMPLIFY) risponde NON SO se B è indefinita,come detto nella seguente osservazione:

Oss 2.4: il valutatore semantico di FOL risponde per formule chiuse non quantificate A:

V se $SS \models^{v} A$ e A è definita

F se $SS \models^{v} \neg A$ e A è definita

NON SO altrimenti (cioè se A non è definita: non $SS \models^{v} A$ e non $SS \models^{v} \neg A$).

A questo punto il nostro interesse si sposta sulle proprietà di $\models^{v}$ per classi di strutture di simulazione; allo scopo diamo la seguente relazione di $\supseteq$ tra strutture di simulazione.

Def 2.8: date due strutture di simulazione SS e SS' diciamo che $SS' \supseteq SS$ sse sono soddisfatte simultaneamente le seguenti condizioni:

1) $D_{SS} = D_{SS'}$;
2) $F_{SS'} \supseteq F_{SS}$ e per ogni $f \in F_{SS}$. $SS'(f) \supseteq SS(f)$
3) $P_{SS'} \supseteq P_{SS}$ e per ogni $p \in P_{SS}$. $SS'(f) \supseteq SS(f)$
4) $C_{SS'} \supseteq C_{SS}$ e per ogni $c \in C_{SS}$. $SS'(c) \supseteq SS(c)$

La prima condizione induce ad una esposizione più semplice dei risultati anche se questi

potranno venire conclusi sotto la condizione più generale $D_{SS'} \supseteq D_{SS}$.

Propr 2.1: $\supseteq$ è riflessiva e transitiva.

Propr 2.2: A denotata in SS $\Rightarrow$ A denotata in ogni SS' $\supseteq$ SS;
A definita in SS $\Rightarrow$ A definita in ogni SS' $\supseteq$ SS.

Rispetto a classi di strutture di simulazione è possibile formalizzare la nozione *non SS* $\models^{\vee}$ *A*, finora lasiata intuitiva, come segue:

Def 2.9: Per ogni fbf A non definita su SS,
non SS $\models^{\vee}$ *A* sse $\forall$SS' $\supseteq$ SS. def(A, SS') $\Rightarrow$ SS' $\models^{\vee}$ A oppure SS' $\models^{\vee}$ $\neg$A

Si osservi che:

- la definizione esplicita ciò che è stato osservato in 2.3, ma questa volta rispetto a tutte le estensioni di SS;
- def 2.9 è in armonia con la definizione di SS $\models^{\vee}$ $\neg$A quando A è definita su SS.

E' possibile provare ora la seguente:

Propr 2.3: SS $\models^{\vee}$ A $\Leftrightarrow$ $\forall$SS' $\supseteq$ SS. SS' $\models^{\vee}$ A.
Dim: per induzione sulla complessità della formula. Si noti che def 2.9 è cruciale per la dimostrazione del passo di induzione relativo a $\rightarrow$. $\otimes$

Def 2.10: Una struttura di simulazione per un linguaggio L è totale sse sono simultaneamente soddisfatte le seguenti condizioni:

1) $F_{SS} = F(L)$, $P_{SS} = P(L)$, $C_{SS} = C(L)$
2) ogni funzione SS(f) (per ogni $f \in F_{SS}$) e ogni predicato SS (p) (per ogni $p \in P_{SS}$) è totale.

Propr 2.4: per ogni struttura totale SS_L, la struttura corrispondente SS_L' costituita da una completa denotazione dei simboli del linguaggio (vedi Oss 2.1) è un modello per L; viceversa, ogni modello per L corrisponde ad una struttura di simulazione totale; se SS_L' è un modello per L allora la nozione di verità $\models^{\vee}$ (definita sopra) collassa con la definizione di verità in un modello.

Propr 2.5: per ogni assegnamento $\underline{x} = \underline{\alpha}$, per ogni struttura di simulazione SS e per ogni fbf A se A è definita in SS per $\underline{x} = \underline{\alpha}$, vale:

$$\forall SS' \supseteq SS \ (SS' \models^{\vee} A \text{ per } \underline{x} = \underline{\alpha}) \Leftrightarrow \forall M \supseteq SS.(M \models\!= A \text{ per } \underline{x} = \underline{\alpha})$$

Dim:

$\Rightarrow$) banale: ogni modello è un struttura di simulazione.

$\Leftarrow$) per assurdo sia: $\forall$M $\supseteq$ SS.(M $\models\!=$ A per $\underline{x} = \underline{\alpha}$) e
$\exists$SS' $\supseteq$ SS (non SS $\models^{\vee}$ A per $\underline{x} = \underline{\alpha}$); essendo A definita in SS per $\underline{x} = \underline{\alpha}$ per la Propr 2.2 A è definita in SS'; dalla Def 2.7, non SS' $\models^{\vee}$ A implica SS' $\models^{\vee}$ $\neg$A; ma allora

(*) $\exists$ M' $\supseteq$ SS' $\supseteq$ SS.M' $\models\!=$ $\neg$A

(dalla Propr 2.3), assurdo. $\otimes$

Oss 2.5: abbiamo riportato la dimostrazione $\Leftarrow$) benchè semplice, per evidenziare che essa dipende dalla (*), proprietà implicata dalla nozione di *A definita*, ma assai più debole; in altre parole, ogni altra nozione di *A definita* che dia luogo alla (*) consente di provare la Propr 2.5.

Dalle proprietà 2.3 and 2.5 segue immediatamente:

Propr 2.6: Se A è definita per $\underline{x} = \underline{\alpha}$, allora SS $\models^{\vee}$ A sse$\forall$M . (M $\supseteq$ SS $\Rightarrow$ M $\models\!=$ A)

Le proprietà 2.5 e 2.6 sono interessanti poichè consentono di connettere $\models^{\vee}$ con la nozione di verità in una classe di modelli o, equivalentemente, di strutture di simulazione.

A questo punto, non volendoci vincolare a particolari definizioni di denotanza, conviene assumere la Propr 2.6 come definizione. Cioè diamo la seguente:

Def 2.11: fissata una struttura di simulazione SS definiamo SS $\models^{*}$ A come segue:

$$SS \models^{*} A \text{ iff } \forall M.(M \supseteq SS \Rightarrow M \models\!= A)$$

Come già osservato, tale definizione si differenzia dalla precedente SS $\models^{\vee}$ A per la mancanza dell'ipotesi di definitezza della formula A rispetto alla struttura di simulazione SS, cioè SS $\models^{*}$ A risulta essere una nozione più generale di $\models^{\vee}$, pur mantenendo gli stessi effetti di SS $\models^{\vee}$ A nei casi in cui è verificata l'ipotesi che A sia definita.

Definiamo ora, cosa si intende per formula logicamente valida su una struttura di simulazione SS.

Def 2.12: $\models^{*}$ A sse $\forall$ SS. SS $\models^{*}$ A.

Può essere dimostrata a questo punto l'equivalenza tra $\models^{*}$ *A* e la nozione di validità classica (la dimostrazione non è riportata perchè priva di difficoltà concettuali):

Th 2.1: $\models\!=$ A iff $\models^{*}$ A.

Pensando ad una teoria T definiamo cosa si intende per conseguenza logica di una formula A in una teoria T rispetto alle strutture di simulazione:

Def 2.13: $T \models^{*}_{=} A$ sse $\forall$ SS. (($\forall \sigma \in T$. SS $\models^{*}_{=} \sigma$) $\Rightarrow$ SS $\models^{*}_{=}$ A).

Analogamente a quanto fatto sopra, sarà possibile dimostrare il seguente teorema di equivalenza tra $T \models^{*}_{=} A$ e la nozione classica di conseguenza logica:

Th 2.2: T $\models_{=}$ A sse T $\models^{*}_{=}$ A

Dim:

$\Leftarrow$) segue direttamente da {SS} $\supseteq$ {M}.

$\Rightarrow$) supponiamo, per assurdo, che: $\neg\forall$ SS. (($\forall \sigma \in T$. SS $\models^{*}_{=} \sigma$) $\Rightarrow$ SS $\models^{*}_{=}$ A); dalla Def 2.11: $\neg\forall$ SS. (($\forall \sigma \in T$. ($\forall$ M $\supseteq$ SS. M $\models_{=} \sigma$)) $\Rightarrow$ ($\forall$ M $\supseteq$ SS M $\models_{=}$ A)); quindi $\exists$ SS. (($\forall \sigma \in T$. ($\forall$ M $\supseteq$ SS. M $\models_{=} \sigma$)) $\wedge$ $\neg$($\forall$ M $\supseteq$ SS M $\models_{=}$ A)); chiamiamo tale struttura SS': (($\forall \sigma \in T$. ($\forall$ M $\supseteq$ SS'. M $\models_{=} \sigma$)) $\wedge$ ($\exists$ M $\supseteq$ SS' M $\models_{=}$ $\neg$A)); otteniamo dunque una struttura di simulazione tale che i modelli che la contengono sono modelli per T su cui vale la formula $\neg$A Dall'ipotesi: $\forall$ M. (($\forall \sigma \in T$. M $\models_{=} \sigma$) $\Rightarrow$ M $\models_{=}$ A); cioè ogni modello per T verifica la formula A. Assurdo. $\otimes$

Concludiamo dando i teoremi di completezza debole e forte come corollari di Th 2.1 e Th 2.2:

Corollario 2.1: (a) $\models^{*}_{=}$ A sse $\vdash$ A

(b) T $\models^{*}_{=}$ A sse T $\vdash$ A

Dim: banale conseguenza di Th 2.1, 2.2 e del teorema di completezza per la logica classica. $\otimes$

2.2 RELAZIONI TRA LA NOZIONE DI DEFINITEZZA E LA NOZIONE CLASSICA DI CONSEGUENZA LOGICA

Per capire le relazioni che esistono tra l'atteggiamento di FOL che fa riferimento ad una nozione di definitezza (il valore di una fbf può essere NON SO) e le nozioni di *val* (Def 2.4 e Def 2.5) e di conseguenza logica derivate da $\models^{*}_{=}$ (ed equivalenti a quelle della logica classica come appena mostrato), risulterà utile evidenziare la dimostrabilità di alcuni corollari. Tali corollari hanno l'obiettivo di mostrare come sia possibile parlare della nozione di formula logicamente valida e del concetto di conseguenza logica per una teoria, passando attraverso

la nozione di definitezza e di $\models^{v}_{=}$, in modo da ottenere un legame tra $\models^{*}_{=}$ e $\models^{v}_{=}$. E' bene precisare che tale legame non deve venir creato necessariamente sfruttando la definizione che è stata data per FOL, espressa dalla Propr 2.5. Come si osserva in Oss 2.5 è possibile infatti ridursi ad un indebolimento della definizione di *A definita* assicurando l'esistenza di un modello che verifica (*). Tale indebolimento soddisferà anche le condizioni espresse dai seguenti corollari:

Corollario 2.2: $\forall$ SS . SS $\models^{*}_{=}$ A sse $\forall$ SS . (def A $\Rightarrow$ SS $\models^{v}_{=}$ A)

Dim:

$\Rightarrow$) poniamo per assurdo che $\neg\forall$ SS . (def A $\Rightarrow$ SS $\models^{v}_{=}$ A; $\exists$ SS . $\neg$(def A $\Rightarrow$ SS $\models^{v}_{=}$ A); cioè $\exists$ SS . (def A $\wedge \neg$SS $\models^{v}_{=}$ A); se la formula A è definita allora dalla def 2.7, $\neg$SS $\models^{v}_{=}$ A $\Rightarrow$ SS $\models^{v}_{=}$ $\neg$A; per Propr 2.6 $\forall$M $\supseteq$ SS. M $\models_{=}$ $\neg$A. Assurdo con la nozione di $\models^{*}_{=}$.

$\Leftarrow$) poniamo per assurdo che $\neg\forall$SS . SS $\models^{*}_{=}$ A; cioè $\neg\forall$SS . ($\forall$M $\supseteq$ SS . M $\models_{=}$ A); quindi $\exists$ SS . ($\exists$ M $\supseteq$ SS . M $\models_{=}$ $\neg$A); chiamiamo tale struttura SS': $\exists$ M $\supseteq$ SS' . M $\models_{=}$ $\neg$A ; per Propr 2.6 se A fosse definita su SS' si avrebbe SS' $\models^{v}_{=}$ $\neg$A; poichè def($\neg$A) implica def(A) per Def 2.6, avremo SS' $\models^{v}_{=}$ A per ipotesi. Assurdo. $\otimes$

Corollario 2.3: $\forall$ SS. (($\forall \sigma \in T$. ($\forall$ M $\supseteq$ SS. M $\models_{=} \sigma$)) $\Rightarrow$ ($\forall$ M $\supseteq$ SS . M $\models_{=}$ A) sse

$\forall$ SS. (($\forall \sigma \in T$ def $\sigma \wedge$ SS $\models^{v}_{=} \sigma$) $\Rightarrow$ (def A $\Rightarrow$ SS $\models^{v}_{=}$ A))

Dim: la dimostrazione è omessa perchè simile alla precedente. $\otimes$

2.3 CONCLUSIONI

Per concludere, pensiamo sia necessario chiarire quali siano state le idee chiave che ci hanno portato a seguire l'approccio semantico precedente, per giungere ad analizzare il comportamento di FOL.

Il nostro primo passo è individuabile nell'analisi della situazione in cui una formula da valutare risulti essere completamente denotata sulla struttura di simulazione presa in considerazione, rispetto ad un fissato assegnamento alle variabili libere. A tale proposito, dobbiamo osservare che in FOL il valutatore lavora solo su fbf chiuse e per tale motivo è stato necessario introdurre i relativi assegnamenti alle variabili libere occorrenti nelle formule trattate. Si è definita, quindi, la nozione di validità, indicata con il simbolo $\models^{v}_{=}$, e la proprietà Propr 2.6., da cui deriva che:

- quando una formula contiene solo simboli denotati nella struttura SS su cui si valuta, la validità della formula o della sua negata si trasmette a tutte le strutture di simulazioni complete (o modelli classici) che contengono SS
- SS' $\models^{v}_{=}$ A iff $\forall$ SS $\supseteq$ SS' . SS $\models^{v}_{=}$ A

Questi due punti mettono in luce che qualsiasi aggiunta di informazione, non contraddittoria con quella già esistente, ci permette di mantenere validi i risultati ottenuti precedentemente.

Per studiare i rapporti con la logica classica, la nozione $/\!\!\underset{\vee}{=}$ doveva essere messa in relazione con la nozione classica $/\!\!=$. Si è dunque cercata una nozione intermedia tra le due e tale da cogliere quegli aspetti che in $/\!\!\underset{\vee}{=}$ venivano soffocati per la presenza di ipotesi di definitezza e che in $/\!\!=$ mancavano, a causa della completa presenza di informazione.

E' stata poi definita in Def 2.11, la nozione indicata col simbolo $/\!\!\underset{*}{=}$ ed è stata utilizzata per dimostrare i teoremi Th 2.1, Th 2.2 e il Corollario 2.1 che stabiliscono in modo chiaro il legame che ci si proponeva.

Questo inquadramento semantico in cui si collocano le strutture di simulazione, permette di chiarire la correttezza dell'uso di tali strutture parziali sostituite ai modelli classici. Tale risultato appoggia, quindi l'impiego della conoscenza semantica nel sistema.

Per quanto concerne le possibili conseguenze di tali risultati, pensiamo che tale analisi possa essere una buona base per valutare in modo preciso il meccanismo di semplificazione sintattico-semantico presente in FOL e studiare le relazioni che occorrono tra le teorie e le strutture parziali.

Il primo proposito è stato informalmente considerato da L.Aiello in (Aiello, 1980). Ma un effettivo studio sull'uso di dimostrazioni miste (cioè sintattiche e semantiche) non è stato pubblicato.

Sul secondo proposito, osserviamo che a causa delle proprietà delle strutture di simulazione, è possibile studiare una classe di teorie che ammettono strutture di simulazione caratterizzanti la classe delle fbf chiuse non quantificate dimostrabili costruttivamente a partire dalla teoria stessa (Carbone, 1985). E' possibile proporre tecniche per la costruzione automatica di questo tipo di strutture di simulazione.

Pensiamo che risultati interessanti possano essere ottenuti sfruttando un approccio Prolog-like o un approccio che considera l'interpretazione costruttiva di prove (Miglioli, Ornaghi, 1981; Miglioli, Ornaghi, 1982). In particolare, il primo approccio permetterà di ottenere una sorta di omogeneità tra il linguaggio logico ed il linguaggio in cui le strutture di simulazione sono espresse.
Il secondo approccio permetterà invece di costruire strutture estraendo informazione da prove costruttive (Miglioli, Ornaghi, 1981).

RINGRAZIAMENTI

Desidero ringraziare Mario Ornaghi e Ugo Moscato dell'Università di Milano per la costante disponibilità e il prezioso aiuto offerti durante lo svolgimento della tesi di laurea che è all'origine di questo lavoro.

BIBLIOGRAFIA

1. Aiello, L., Evaluating function defined in first order logic or in defence of the semantic attachment in Fol. Stanford Artificial Intelligence Laboratory, Stanford University, Stanford, 1980.
2. Bencivenga, E., (ed.), Le logiche libere. (Boringhieri, Torino, 1976).
3. Bencivenga, E., Una logica dei termini singolari. (Boringhieri, Torino, 1980).
4. Carbone, A., Un'analisi di alcuni aspetti logici e semantici del Fol. Tesi di laurea, Università di Milano, a.a.1984/85.
5. Chang, C.C., Keisler, H.J., Model theory. (North Holland, Amsterdam, 1973).
6. Lukasiewiecz, J., O logice trojwartosciowej. *Ruch Filozoficzny* , 5 (1920).
7. Lukasiewiecz, J., Philosophisch Bemerkungen zu den mehrwertigen Systemem des Aussagenkalkuls. *Monatshefte fur Mathematik und Phisik*, XXXVII, (1930) 51-77.
8. Miglioli,P., Ornaghi,M., A logically justified model of computation. *Fundamenta Informaticae* IV.1 and IV.2 (1981).
9. Miglioli,P., Ornaghi,M.,Constructive proofs and logical computation. *Pocitace a Umela Inteligencia*, (1982).
10. Prawitz, D., Natural Deduction - A proof theoretical study. (Almquist & Wiksell, Stockholm, 1965).
11. Van Fraassen, B.C., The completeness of free logics, *Zeitschrift Fur Mathematische Logik und Grundlagen Der Mathematik*, vol. 12 (1966a) 245-254.
12. Van Fraassen, B.C., Singular Terms, Truth-Value Gaps, and Free Logic, *Journal of Philosophy*, vol.67 (1966b) 481-495.
13. Van Fraassen, B.C., A topological proof of the Lowenheim-Skolem, Compactness, and Strong Completeness Theorems for Free Logic, *Zeitschrift Fur Mathematische Logik und Grundlagen Der Mathematik*, vol. 14 (1968) 245-254.
14. Weyhrauch, R.W., A user manual for Fol. Stanford Artificial Intelligence Laboratory, Memo AIM 235.1, Stanford University, Stanford (1977).
15. Weyhrauch, R.W., Prolegomena to a theory of mechanized formal reasoning. *Artificial Intelligence* 13 (1980) 133-170.