Specifiche

Specifiche = descrizione delle caratteristiche di un prodotto

Esempio non informatico: la specifica di un detersivo è data da quale quantità è necessaria per fare un bucato, quanto pulisce alle varie temperature, per quali tessuti può essere usato, etc.

Esempio informatico: specifiche funzionali di un programma: su quali piattaforme può girare, di quanta memoria ha bisogno, che cosa fa.

Le specifiche prescindono da come il prodotto è realizzato, ma sono espresse in termini relativi al dominio di applicazione del prodotto:

- necessità di astrazione (per semplicità e quindi comprensibilità)
- leggibilità da parte dell'utente che di solito non ha le competenze per capire una descrizione tecnica del prodotto
- segreto industriale (ovvie ragioni di mercato suggeriscono di non divulgare come è realizzato un prodotto)

Nel caso del detersivo non si può/deve usare la formula chimica corrispondente.

Nel caso del SW non si può usare il codice.

Specifiche Formali

Consideriamo il caso del SW

Il linguaggio naturale, che è quello che tutti conoscono è comodo da usare, ma troppo ambiguo per un contratto fra fornitore e utente

I linguaggi formali, che sono assolutamente non ambigui, sono artificiali e spesso o troppo poveri da un punto di vista espressivo, o troppo difficili da capire e padroneggiare.

Soluzione ideale (dreaming):

Avere un “traduttore automatico” da linguaggio naturale a linguaggio formale.

Essendo il linguaggio naturale ambiguo e quello formale no, è impossibile realizzarlo.

Esistono soluzioni di compromesso, cioè linguaggi semiformali, o linguaggi formali con rappresentazioni intuitive (eg grafiche) che trattano aspetti ristretti del linguaggio.

Specifiche Funzionali

A seconda degli aspetti che si vogliono discutere i diversi linguaggi risultano più o meno adeguati.

Requisiti “imperativi”…logiche tipo alla Hoare

Requisiti temporali (soprattutto per paradigmi concorrenti)…logiche temporali dove ci sono connettivi tipo “prima o poi”, “d’ora innanzi sempre”, “questo finché non diventa vero quello”

Requisiti “funzionali”, riguardano i servizi offerti dal sw, eg quali procedure comprende, con quali tipi e che calcolano quale funzione.

Inizialmente (fine anni 60, inizio 70) legati alla descrizione dei tipi di dato (= famiglie di insiemi + operazioni primitive per manipolarli, eg stack+elem+empty+pop+top+push+length).

Pare stiano tornando di moda per descrivere le interfacce delle componenti

Specifiche Algebrico/Logico

Servono a descrivere requisiti funzionali di un sistema

Consideriamo il caso puramente funzionale (si possono estendere per gestire cosi con stato, ma il concetto di stato non è primitivo)

Gli ingredienti sono

- Sintassi dell'interfaccia (segnatura, italianizzazione di signature)
- Implementazione, o realizzazione (algebra, o tipo di dato concreto)
- Specifica, o descrizione logica di proprietà (assiomi, o formule)

Dare una specifica vuol dire fissare i nomi dei tipi e delle funzioni/procedure che il prodotto è in grado di fornire e fissare attraverso delle formule logiche le proprietà astratte dei tipi e delle funzioni stesse (cioè proprietà sulla semantica).

Data una specifica ci sono tre processi fondamentali per utilizzarla:

- derivazione di proprietà (una formula è deducibile dagli assiomi?)

- validazione (un'implementazione soddisfa gli assiomi?)

- prototipizzazione (si deriva, ove possibile, una implementazione)

Algebre Parziali con Predicati

Per trattare il sw ad un alto livello di astrazione è conveniente modificare le algebre (come viste ad Algebra, spero) e/o la logica del prim'ordine (vista a Logica) e definire le algebre parziali con predicati.

Una segnatura $\Sigma = (S,F,P)$ consiste di

- Un insieme S di nomi dei tipi, o sort
- Una famiglia F di simboli di funzioni, indiciata su $S^* \times S$.

  se $f \in F_{s_1 \ldots s_n, s}$ indicheremo $f: s_1 \times \ldots \times s_n \to s$
- Una famiglia P di simboli di predicati, indiciata su S^*.

  se $p \in P_{s_1 \ldots s_n}$ indicheremo $p: s_1 \times \ldots \times s_n$

Esempio una segnatura per le liste di interi Σ_{list}:

S = {int,list}

F consiste di 0: $\to$ int, S: int $\to$ int, empty: $\to$ list, push: int $\times$ list $\to$ list, pop: list $\to$ list, top: list $\to$ int

P consiste di is0: int, isempty: list, isIn: int $\times$ list

Algebre Parziali con Predicati (2)

Data una segnatura $\Sigma = (S,F,P)$ una Σ-algebra (parziale con predicati) consiste di

- Un insieme s^A per ogni $s \in S$, detto carrier o supporto di tipo s in A
- Una funzione parziale $f^A: s_1^A \times \ldots \times s_n^A \to s^A$ per ogni $f: s_1 \times \ldots \times s_n \to s \in F$, detta interpretazione di f in A
- Un insieme $p^A \subseteq s_1^A \times \ldots \times s_n^A$ per ogni $p: s_1 \times \ldots \times s_n \in P$, detto insieme di verità di p in A
- Se tutte le funzioni di un'algebra sono totali, l'algebra si dice totale

Esempio un'algebra sulla segnatura Σ_{list}:

$int^A = \mathbb{N}$, $list^A = \mathbb{N}^*$

$0^A = 0$, $S^A(x) = x+1$, $empty^A = \emptyset$, $push^A(n,x) = nx$, $pop^A(nx) = x$ e $pop^A(\emptyset)$ indefinito, $top^A(nx) = n$ e $top^A(\emptyset)$ indefinito.

$is0^A = \{0\}$, $isempty^A = \{\emptyset\}$, $isIn^A = \{(n,x) \mid \exists i \in [1,|x|].x(i)=n\}$

Logiche Parziali

Data una segnatura Σ = (S,F,P) i Σ-termini sono definiti come per la segnatura (S,F).

Le formule logiche sono definite come nel caso della logica del prim'ordine sui simboli funzionali F e predicativi P, ovvero per induzione come segue:

- $p(t_1,\ldots,t_n)$ è una formula atomica per ogni p: $s_1 \times \ldots \times s_n \to P$, e termini t_i di tipo s_i
- Se φ e ψ sono formule, allora tali sono anche $\neg\varphi$, $\varphi \wedge \psi$, $\varphi \vee \psi$, $\varphi \Rightarrow \psi$, $\exists x{:}s.\varphi$ e $\forall x{:}s.\varphi$

Esempi di formule sulla segnatura Σ_{list}:

isempty(pop(push(n,empty)))

$\forall$x: list. Isempty(x) $\Rightarrow$ isIn(top(x),x)

$\exists$n:int. is0(n)

$\forall$ n:int. is0(n) $\Rightarrow$ n **=** 0

NO quest'ultima non è (per il momento) una formula perché compare **=**

Interpretazione e validità

La validità delle formule logiche in un'algebra parziale con predicati è definita come nel caso della logica classica, per induzione sulla forma delle formule, basandosi sul concetto di interpretazione $t^{A,V}$ di un termine t in un'algebra A rispetto ad una valutazione V (almeno) delle variabili che vi compaiono in A:

Se V: X→ A è una valutazione delle variabili libere di φ, allora φ vale (o è valida) in A rispetto a V, indicato $A \models_V \varphi$ se e solo se:

- se φ è $p(t_1,\ldots,t_n)$ e $t_1^{A,V},\ldots,t_n^{A,V} \in p^A$
- se φ è ¬ψ e $A \not\models_V \psi$;
- se φ è ψ∧χ e sia $A \models_V \psi$ che $A \models_V \chi$;
- se φ è ψ ∨ χ e $A \models_V \psi$ oppure $A \models_V \chi$;
- se φ è ψ → χ e $A \models_V \psi$ oppure $A \not\models_V \chi$;
- se φ è ∃x:s.ψ ed esiste V':X→{x}→ A t.c. V'(y) =V(y) se y≠x e $A \models_{V'} \psi$;
- se φ è ∀x:s.ψ e per ogni V': X→{x}→ A t.c. V'(y) = V(y) se y≠x e $A \models_{V'} \psi$;

Per poter fare un esempio bisogna prima definire l'interpretazione $t^{A,V}$ (e lo faremo nel prossimo lucido)

Una formula φ è valida in un'algebra A, indicato A⊨ φ, se e solo se $A \models_V \varphi$ per ogni valutazione delle variabili libere di φ in A,

cioè se e solo se $A \models_V \{\forall x:s \mid x \in FreeVar(\varphi)_s\}.\ \varphi$ (chiusura universale di φ).

Interpretazione Parziale

Consideriamo termini costruiti su una segnatura $\Sigma = (S,F,P)$ e una famiglia X indiciata su S di insiemi di variabili.

La famiglia di tali termini si indica $T_\Sigma(X)$.

Una valutazione $V: X \to A$ è una famiglia di funzioni totali $V_s: X_s \to s^A$

Data una valutazione $V: X \to A$, l' interpretazione dei termini, indicata $\_^{A,V}: T_\Sigma(X) \to A$, è una famiglia di funzioni parziali $\_^{A,V}: T_\Sigma(X)_s \to s^A$ definita per induzione sui termini come segue

- $x^{A,V} = V_s(x)$ se $x \in X_s$
- se $t_1^{A,V} = a_1 \in s_1^A, \ldots, t_n^{A,V} = a_n \in s_n^A$ e $f^A(a_1,\ldots,a_n) = a \in s^A$, allora $f(t_1,\ldots,t_n)^{A,V} = a$

devono esserlo tutti i suoi sottotermini

e i rispettivi valori devono appartenere al dominio di f

perché un termine sia definito

Esempio (solita segnatura ed algebra)

$(pop(push(n,empty)))^{A,V} =$

$pop^A((push(n,empty))^{A,V}) =$

$pop^A(push^A((n)^{A,V},(empty)^{A,V})) =$

$pop^A(push^A(V(n),empty^A)) =$

$pop^A(push^A(V(n),\emptyset)) = pop^A(V(n)\emptyset) = \emptyset$

Da cui per qualsiasi valutazione V

$A \models_V isempty(pop(push(n,empty)))$

perché

$(pop(push(n,empty)))^{A,V} = \emptyset \in isempty^A = \{\emptyset\}$,

Uguaglianza e Parzialità

Vogliamo aggiungere il simbolo di uguaglianza alla nostra logica.

Per poter esprimere formule di qualsiasi tipo con uguaglianze, basta aggiungere l'uguaglianza come formula atomica e darne l'interpretazione semantica in una generica algebra rispetto ad una valutazione delle formule.

Poi essendo sia le formule che la loro validità definita per induzione, si propagheranno senza problemi.

Nel caso totale due termini sono uguali se denotano (sono rappresentazioni sintattiche di) lo stesso valore, ovvero se la loro valutazione porta allo stesso risultato.

Nel caso parziale, le due frasi non sono più equivalenti perché c'è il caso in cui le due valutazioni portano allo stesso risultato: indefinito.

Quindi nel caso parziale ci sono naturalmente due nozioni di uguaglianza

- $t = t'$ (uguaglianza forte) $A \models_V t=t'$ se e solo se $t^{A,V} =_{\square} t'^{A,V}$ (eventualmente indefiniti entrambi)
- $t =_e t'$ (uguaglianza esistenziale) $A \models_V t =_e t'$ se e solo se $\square a \square s^A.t^{A,V} = a = t'^{A,V}$

Definitezza e Strettezza

Abbreviazione: Def(t) sta per $t =_e t$ (la sua validità vuol dire che $t^{A,V} \in s^A$,

t è definito in A)

Alternativamente avremmo potuto aggiungere un predicato unario Def con interpretazione fissa e coincidente con tutto il carrier

(fissata, volendo dall'assioma $\forall x{:}s.Def(x)$, uno per ogni tipo s in S)

Funzioni (e predicati) in quest'approccio sono stretti, cioè restituiscono un valore (sono veri) solo su argomenti definiti.

Prop. Fissate una segnatura $\Sigma = (S,F,P)$, una S-famiglia X di variabili, termini $t_i \in T_\Sigma(X)_{s_i}$, una Σ-algebra A e una valutazione $V: X \to A$.

1 Per ogni f: $s_1 \times \ldots \times s_n \to s \in F$, $A \models_V Def(f(t_1,\ldots,t_n))$ implica $A \models_V Def(t_i)$

2 Per ogni p: $s_1 \times \ldots \times s_n \in P$, $A \models_V p(t_1,\ldots,t_n)$ implica $A \models_V Def(t_i)$

Prova

1 $A \models_V Def(f(t_1,\ldots,t_n))$ se e solo se $f(t_1,\ldots,t_n)^{A,V} \in s^A$ cioè, per definizione di interpretazione, se e solo se $t_1^{A,V} = a_1 \in s_1^A,\ldots,t_n^{A,V} = a_n \in s_n^A$ e $f(t_1,\ldots,t_n)^{A,V} = f^A(a_1,\ldots,a_n) \in s^A$, allora $t_i^{A,V} \in s_i^A$ e quindi $A \models_V Def(t_i)$.

2 Analogamente (per esercizio)

Quindi l'uguaglianza forte (che vale anche quando entrambi i lati non sono definiti) non potrà mai essere un predicato, mentre l'uguaglianza esistenziale sì.

Logiche a 2/3 valori

Nel definire la validità di una formula abbiamo scelto di allontanarci il meno possibile dalla logica classica (totale).

Per questo non abbiamo cambiato il dominio di valutazione delle formule, che possono essere solo vere o false.

Quindi abbiamo scelto che un'applicazione di predicato ad un termine indefinito valesse falso.

(Attenzione: questo non vuol dire che tutte le formule in cui compare un termine indefinito sono false, eg $A \models\neq_V Def(t)$ implica $A \models=_V \neg Def(t)$)

Una scelta più radicale sarebbe stata passare ad una logica a 3 valori, in cui una formula può essere vera, falsa o indefinita.

Logiche a 3 valori permettono di discriminare situazioni che le logiche a 2 valori inevitabilmente identificano.

Però ai fini delle specifiche del sw funzionale e sequenziale, non sono strettamente necessarie (e quindi non le facciamo)

La logica parziale del prim'ordine ha lo stesso potere espressivo di quella totale.

Specifiche

Una specifica Sp è una coppia (Σ,Ax), dove Ax è un insieme di Σ-formule, dette assiomi di Sp.

I modelli di Sp sono tutte le Σ-algebre che soddisfano tutti gli assiomi di Sp, cioè Mod(Sp) = {A | A$\in$ Alg(Σ) e A$\models\varphi$ per ogni $\varphi\in$ Ax}

Fra questo sono particolarmente i modelli term-generated (generati dai termini), cioè quei modelli in cui ogni elemento dei carrier è interpretazione di un termine senza variabili.

Infatti, corrispondono a implementazioni minimali dove ci sono solo quegli elementi che sono indispensabili per poter dare un'interpretazione alle "chiamate" ricevute attraverso l'interfaccia pubblica (cioè la segnatura).

GMod(Sp) = {A | A$\in$ Mod(Sp) e $\_^{A,\emptyset}$: $T_\Sigma(\emptyset)\to$ A surgettiva}

Esempio di specifica sulla segnatura Σ_{list}: $Sp_{list}=(\Sigma_{lis},Ax)$ dove Ax consiste di

Def(empty) $\wedge$ Def(push(n,x)) (cioè push deve essere interpretata da una funzione totale)

pop(push(n,x)) = x $\wedge$ top(push(n,x)) = n

is0(n) $\quad$ n = 0 $\wedge$ n = 0 $\quad$ is0(n)

isempty(x) $\quad$ x = empty $\wedge$ x = empty $\quad$ isempty(x)

isIn(n,push(n,x)) $\wedge\wedge$ isIn(n,empty)

isIn(m, x) $\quad$ isIn(m,push(n,x))

Validazione

Nel formalismo introdotto validare una implementazione rispetto a certi requisiti diventa, formalmente, verificare che un'algebra (l'implementazione) è un modello della specifica (che formalizza i requisiti).

Questo è molto diverso da validare del sw mediante testing, perché è una prova di correttezza in tutti i casi possibili.

Per questa ragione, i sistemi safety critical sono di solito specificati e validati (esempio canonico: il sw embedded in lavatrici, contatori della luce etc)

Data una specifica Sp = (Σ,Ax), ed una Σ-algebra A come si fa a verificare se A è un modello di Sp?

Basta usare la definizione.

Si prende in esame un assioma φ alla volta. isIn(m, x) isIn(m,push(n,x))

Si determinano le sue variabili libere FreeVar(φ). FreeVar(φ) = {m:int,x:list}

Si fissa una valutazione V per FreeVar(φ) in A. Siano V(m)$\in\mathbb{N}$ e V(x)$\in\mathbb{N}^*$

Si applica la definizione di validità e di interpretazione un passo alla volta.

$A \models_V$ isIn(m, x) isIn(m,push(n,x)) sse $A \not\models_V$ isIn(m, x) o $A \models_V$ isIn(m,push(n,x))

$A \not\models_V$ isIn(m, x) sse $(m^{A,V}, x^{A,V}) \notin isIn^A$ sse (V(m), V(x))$\notin isIn^A$ sse per ogni i $\in$[1,|V(x)|].V(x)(i) $\neq$n….etc, etc...

Modelli Rappresentativi

Fra tutti i modelli di una specifica ci interessa individuarne uno che rappresenti, per quanto possibile, tutta la classe dei modelli.

Visto che siamo focalizzati sull'uso della logica, questo vuol dire un modello che ci dia informazioni sulla validità di formule anche negli altri modelli.

Se possibile, ci piacerebbe un modello B (per Best) tale che

* $B \models \varphi$ implica $A \models \varphi$ per ogni modello A e per ogni formula φ.

Supponiamo che esista un tale modello. Allora per ogni termine (senza variabili) t si avrebbe

$B \models Def(t)$ implica $A \models Def(t)$, $B \models \neg Def(t)$ implica $A \models \neg Def(t)$ e sicuramente una delle due premesse è vera.

Quindi in tutti i modelli sarebbero definiti esattamente gli stessi termini.

Se ripetiamo il ragionamento per l'uguaglianza e per le applicazioni di predicato, otteniamo in tutti i modelli valgono le stesse identità e i predicati sono interpretati allo stesso modo, cioè che tutti i modelli sono indistinguibili dal punto di vista della logica.

(Nomenclatura: una specifica i cui modelli sono tutti isomorfi fra loro si dice monomorfa)

Il problema nasce dal fatto che richiedere * per tutte le formule (incluse le negazioni) è troppo forte.

Modello Iniziale

Il problema che è emerso dal lucido precedente è la negazione.

Escludere le formule che contengono un not non è però sufficiente, perché problematiche analoghe nascono dai not "nascosti"

(eg. $\neg\ \ \neg \equiv \neg\neg\ \ \neg$, $t=t' \equiv t=_e t'\quad (\neg Def(t) \vee \neg Def(t')))$

Restringiamo drasticamente il tipo di formule ai soli atomi positivi cioè applicazioni di predicati e uguaglianze esistenziali, che sono sufficienti per caratterizzare un'algebra.

Definizione

Un'algebra I è iniziale per una specifica Sp se e solo se

- È un modello di Sp
- È term-generated

$\Big\}\ I \in GMod(Sp)$

- $I \models \varphi$ implica $A \models \varphi$ per ogni $A \in Mod(Sp)$ e ogni $\varphi \in PAtom(\Sigma)$

  Dove $PAtom(\Sigma)=\{t=_e t' \mid t,t' \in T_\Sigma\} \cup \{p(t_1,\ldots,t_n) \mid t_i \in T_{\Sigma s_i}\}$

Si dice anche che I soddisfa il minimo vero.

Ma una tale I esiste sempre?

In generale no. Vedremo un caso particolare in cui esiste e si può calcolare

Congruenze e Quozienti

Visto che vogliamo costruire un modello term-generated cerchiamo di costruire i carrier come quozienti di termini.

Siccome poi dobbiamo dotare i carrier di struttura algebrica per fare il quoziente non ci basta una qualsiasi relazione d'ordine.

Una famiglia ~ di relazioni $\sim_s \subseteq \mathcal{T}_\Sigma(X)_s \times \mathcal{T}_\Sigma(X)_s$ è una congruenza (caso particolare di una definizione più generale) se

- Ciascuna $\sim_s$ è simmetrica e transitiva (ma non necessariamente riflessiva)
- $t_i \sim_{s_i} t'_i$ e $f(t_1,\ldots,t_n) \sim_s f(t_1,\ldots,t_n)$ implica $f(t_1,\ldots,t_n) \sim_s f(t'_1,\ldots,t'_n)$

Data una congruenza si definisce il quoziente.

$s^\sim = \mathcal{T}_\Sigma(X)_s/\sim$

$f^\sim([t_1],\ldots,[t_n]) = [f(t_1,\ldots,t_n)]$ se $f(t_1,\ldots,t_n) \sim f(t_1,\ldots,t_n)$, indefinita altrimenti

È ben definito?

Sì perché simmetria e transitività permettono di parlare di classi di equivalenza e la proprietà sulle funzioni garantisce la correttezza della definizione di $f^\sim$.

Costruzione del modello iniziale

Sia $\sim^{Sp} = \{\sim^{Sp}_s \subseteq \Sigma_s \times \Sigma_s \mid s \in S\}$ definita da

$t \sim^{Sp} t'$ se e solo se per ogni modello A di Sp si ha $A \models t =_e t'$

Allora $\sim^{Sp}$ è una congruenza (verifica per esercizio)

Sia I(Sp) l'algebra parziale coi predicati definita da:

- $s^{I(Sp)} = \Sigma_s / \sim^{Sp}$
- $f^{I(Sp)}([t_1],\ldots,[t_n]) = [f(t_1,\ldots,t_n)]$ se $f(t_1,\ldots,t_n) \sim^{Sp} f(t_1,\ldots,t_n)$, indefinita altrimenti
- $p^{I(Sp)} = \{([t_1],\ldots,[t_n]) \mid A \models p(t_1,\ldots,t_n)$ per ogni $A \in Mod(Sp)\}$

1. I(Sp) è ben definita
2. Se $I(Sp) \in Mod(Sp)$ allora è il modello iniziale di Sp

- La prova di 1 è banale (per esercizio)
- La prova di 2 si basa sul (=è banale, per esercizio, assumendo che sia vero) lemma seguente

3. Data una valutazione $V: X \to I(Sp)$, $I(Sp) \models_V \varphi$ se e solo se

   $I(Sp) \models_V \varphi[t_x/x \mid x \in X]$ dove $t_x \in V(x)$

Esistenza del modello iniziale

In generale l'algebra I(Sp)non è un modello:

Sp =(Σ,Ax) dove Σ = ({s},{a,b},$\emptyset$) e Ax={Def(a) Def(b)}

Ha un modello in cui è definita a ma non b e viceversa un modello in cui è definita b ma non a, per cui in I(Sp) entrambe non sono definite e quindi non è un modello

Teorema (solo enunciato)

Se Sp è una specifica positive conditional, allora ammetto modello iniziale.

Dove una specifica è positive conditional se e solo se è logicamente equivalente (=hanno gli stessi modelli) ad una i cui assiomi lo sono tutti e un assioma è positive conditional se e solo se è della forma

$\Phi_1 \Phi \ldots \Phi \Phi_n \quad \Phi_0$ dove ogni Φ_i è un atomo e se Φ_i è una uguaglianza forte t=t' per qualche i$\in$[1,n], allora esiste j$\in$[1,n] tale che Φ_j è un atomo positivo e o t o t' compare in Φ_j (guarded strong equality)

Le specifiche p.c. sono particolarmente interessanti perché corrispondono metodologicamente a definizioni ricorsive (induttive)

e soprattutto perché esiste un sistema automatico per costruire I(Sp)

(è semidecidibile se $t \sim^{Sp} t'$)

Sistemi deduttivi

Un sistema deduttivo è una relazione fra insiemi di formule Γ (gli assiomi assunti veri) e formule Δ (le formule dedotte a partire dagli assiomi) indicato $\Gamma \vdash \Delta$

Data una nozione di validità $\models$, un particolare sistema deduttivo è $\models$, definito da $\Gamma \models \Delta$ se e solo se $A \models \Gamma$ implica $A \models \Delta$ per ogni modello A.

Un sistema deduttivo è sound (corretto) se $\Gamma \vdash \Delta$ implica $\Gamma \models \Delta$.

Un sistema deduttivo è complete (completo) rispetto ad una classe di formule Φ se $\Gamma \models \Delta$ implica $\Gamma \vdash \Delta$ per ogni Γ, Δ, Φ

Perché abbia un'utilità pratica un sistema deduttivo deve essere:

- Sound (non vogliamo dedurre falsità)
- Complete rispetto ad una classe ragionevolmente ampia di formule
- Descritto in modo effettivo, cioè in modo da poter essere usato come mezzo di calcolo

Di solito un sistema deduttivo si presenta mediante un insieme di regole di inferenza.

Sistema di Birkhoff

Il sistema di Birkhoff è un sistema deduttivo per la logica equazionale omogenea e totale.

Assiomi propri $\dfrac{}{\varphi}$ $\varphi \in Ax$

Equivalenza

$$\frac{}{t=t} \quad \text{riflessività} \qquad \frac{t=t'}{t'=t} \quad \text{simmetria} \qquad \frac{t=t' \quad t'=t''}{t=t''} \quad \text{transitività}$$

Congruenza

$$\frac{t_1 = t'_1, \ldots, t_n = t'_n}{f(t_1,\ldots,t_n) = f(t'_1,\ldots,t'_n)} \qquad \frac{p(t_1,\ldots,t_n) \quad t_1 = t'_1, \ldots, t_n = t'_n}{p(t'_1,\ldots,t'_n)}$$

Istanziazione $\dfrac{\varphi}{\varphi[V]}$ $\varphi \in Form(\Omega,X)$, $V:X \to T_\Omega(Y)$

Vogliamo estenderlo al caso parziale eterogeneo e condizionale.

Cioè vogliamo che gli assiomi propri ed eventualmente passi intermedi della computazione siano della forma

$\varphi_1 \wedge \ldots \wedge \varphi_n \Rightarrow \varphi$ dove tutti gli φ_i sono atomi (cioè uguaglianze forti o atomi positivi)

Sistema di Birkhoff generalizzato

Il fatto di avere formule condizionali richiede di aggiungere una regola di *modus ponens*

$$\frac{\alpha_1 \wedge \ldots \wedge \alpha_n \rightarrow \quad \alpha_i}{\alpha_1 \wedge \ldots \wedge \alpha_{i-1} \wedge \alpha_{i+1} \wedge \ldots \wedge \alpha_n \rightarrow}$$

Avendo parzialità ci sono due nozioni di uguaglianza, la forte e l'esistenziale.

L'uguaglianza forte è riflessiva, mentre quella esistenziale no.

Entrambe sono simmetriche e transitive.

Quindi alle regole del sistema di Birkhoff (che valgono per l'= forte) aggiungiamo

$$\frac{t =_e t'}{t' =_e t} \qquad \frac{t =_e t' \quad t' =_e t''}{t =_e t''} \qquad \frac{t =_e t'}{t = t'} \qquad \frac{t =_e t \quad t = t'}{t =_e t'} \qquad \frac{t =_e t \quad t = t'}{t' =_e t'}$$

simmetria transitività Relazioni fra = e $=_e$

Strettezza $\dfrac{p(t_1,\ldots,t_n)}{t_i =_e t_i}$ $\dfrac{f(t_1,\ldots,t_n) =_e f(t_1,\ldots,t_n)}{t_i =_e t_i}$

Bisogna ancora dire che le variabili sono sempre definite e che (quindi) si possono sostituire solo con termini definiti (quindi la regola di istanziazione va modificata)

Definitezza delle variabili $\dfrac{}{x =_e x}$ $x \in X$ Istanziazione $\dfrac{t =_e t \quad \alpha}{\alpha[t/x]}$

Soundness

Ma il sistema di Birkhoff generalizzato è sound e completo rispetto agli atomi positivi ground (=senza variabili)?

Se sì lo possiamo usare come "algoritmo" di calcolo per $\sim^{Sp}$.

Nella forma data NON è sound.

Controesempio: sia $\Box = (\{s\},\{a,b\},\emptyset)$ l'applicazione della regole di transitività

$$\frac{a=_e x \quad x=_e b}{a=_e b}$$

non è sound.

Se si considera il caso dell'algebra vuota (carrier di sort s insieme vuoto, a e b indefinite entrambe), non ci sono valutazioni per la variabile x, quindi le premesse sono valide, mentre la conseguenza non lo è.

Il punto cruciale è che $a =_e x$ e $x =_e b$ in realtà sono $\Box x{:}s.\ a =_e x$ e $\Box x{:}s.\ x =_e b$ da cui si può dedurre correttamente $\Box x{:}s.\ a =_e b$.

Le soluzioni possibili a questo problema sono:

1 escludere i modelli con carrier vuoto ←

2 quantificare universalmente le variabili in maniera esplicita

3 modificare le regole in modo da non diminuire mai le variabili tranne che con l'istanziazione

Completeness

Il sistema di Birkhoff generalizzato è completo rispetto agli atomi positivi ground (=senza variabili).

In realtà in una forma lievemente diversa è completo anche per gli assiomi della forma $x_1 =_e x_1 \wedge \ldots \wedge x_n =_e x_n \quad \Rightarrow$ con $\Rightarrow$ un atomo positivo.

Per ottenere un sistema completa rispetto a formule del tipo $x_1 =_e x_1 \wedge \ldots \wedge x_n =_e x_n \quad \Rightarrow$ con $\Rightarrow$ un atomo qualunque invece bisogna lavorare un po'.

Sistemi completi per altre classi di formule sono poco studiati nel contesto delle specifiche algebriche perché non hanno applicazioni pratiche.