Appunti delle lezioni del corso "Semantica dei linguaggi di programmazione"

Prof.ssa Anna Labella

A.A. 2005/2006

a cura di Flavio Calva e Vincenzo Corcione

Indice

Introduzione **vii**

I Aspetti introduttivi **1**

1 Semantica dei linguaggi di programmazione **3**

1.1 Semantica statica 4

1.2 Semantica dinamica 4

1.2.1 Semantica operazionale 4

1.2.2 Semantica assiomatica 5

1.2.3 Semantica denotazionale 6

2 I numeri naturali **11**

2.1 L'assiomatizzazione di Campano da Novara 11

2.2 Principio di induzione 13

2.3 Ordinamento 15

2.4 Insiemi Ordinati 18

3 Strutture algebriche **21**

3.1 Reticolo algebrico 21

3.2 Algebra di Boole 23

4 Algebra dei linguaggi **25**

4.1 Introduzione 25

4.2 Definizioni 26

5 Monoidi **29**

II Teoria dei dominî **35**

6 Continuità **37**

6.1 CPO 37

6.2 Dominî 39

6.3 Operazioni tra dominî 40

6.4 Dominî di funzioni 51

6.5 Evaluation e Currying 52

6.6 Punto fisso e minimo punto prefisso 54

6.6.1 Teorema del punto fisso di Tarski 56

7 Ancora sull'algebra dei linguaggi **61**

8 Principio di induzione di Scott sui punti fissi **69**

III PCF **79**

9 PCF: Programming Computable Function **81**

9.1 Termini e tipi 82

9.2 Variabili libere, variabili vincolate e sostituzione 83

9.3 Tipi 84

9.4 Valutazione 87

9.5 Equivalenza contestuale e uguaglianza nella denotazione 90

10 Semantica denotazionale **95**

10.1 Denotazione di tipi 95

10.2 Denotazione di termini 95

10.2.1 Semantica denotazionale di termini PCF 96

10.2.2 La composizione preserva la continuità 98

10.3 Composizionalità 100

10.4 Correttezza 102

11 Adeguatezza e approssimazione formale **105**

11.1 Approssimazione formale di relazioni 105

11.2 Proprietà fondamentali delle relazioni 106

11.3 Dimostrazione proprietà fondamentale di $\triangleleft$ 107
11.4 Estensionalità 110

12 Astrazione completa **113**
12.1 Fallimento dell'astrazione completa 113

IV Esercizi **117**

13 Esercizi **119**
13.1 Esercizio sui dominî di funzione 119
13.2 Esercizio sulla costruzione dei dominî 120
13.3 Esercizio sul PCF 121
13.4 Esercizio valutazione PCF 121
13.5 Esercizio sui dominî 124

Tabella delle abbreviazioni **127**

Tabella delle abbreviazioni **127**

Bibliografia **129**

Indice analitico **129**

Introduzione

Il corso si propone di esaminare, da un punto di vista algebrico, alcuni concetti che sono alla base dell'informatica.

Informatica e matematica

L'informatica inizialmente si è sviluppata a partire dai risultati dell'*analisi numerica*, che ha fornito dei metodi per il calcolo. Successivamente, con lo sviluppo dei *linguaggi artificiali*, essa si è evoluta verso un livello crescente di astrazione, avvicinandosi sempre di più ad una descrizione algebrica.

L'algebra si propone di studiare la "teoria delle operazioni"; noi affiancheremo a questa lo studio di una "teoria dell'ordine".

Tale classificazione delle teorie matematiche è stata definita dalla prestigiosa scuola matematica *Nicolas Bourbaki*, la quale, dopo attente revisioni, propose la suddivisione della matematica in tre filoni principali:

- teoria delle operazioni
- teoria dell'ordine
- topologia

La *teoria delle operazioni* si occupa di *teoria degli insiemi* e di *strutture algebriche* (e.g. gruppi, anelli, campi, ...); la *teoria dell'ordine* tratta gli insiemi *ordinati* o *parzialmente ordinati*; la *topologia* ha forti legami con le prime due teorie e.g. gli aperti di uno spazio topologico sono dei particolari sottoinsiemi dello spazio e quindi si comportano come le parti di un insieme sulle quali si possono definire delle operazioni che inducono topologie e viceversa. Tuttavia, la topologia presenta delle peculiarità che la differenziano dalle due precedenti teorie.

Tabella 1: Nicolas Bourbaki

Tratto da [Wik06].
Nicolas Bourbaki è lo pseudonimo con il quale, a partire dal 1935 e sostanzialmente fino al 1983, un gruppo di matematici di alto profilo, in maggioranza francesi, ha scritto una serie di libri per l'esposizione sistematica di nozioni della matematica moderna avanzata. Con questa operazione scientifica il gruppo si è posto l'obiettivo di *fondare l'intera matematica sulla teoria degli insiemi* attraverso testi che fossero il più possibile rigorosi e generali. Nel corso di questa attività sono stati introdotti nuovi termini e nuovi concetti che hanno avuto una influenza complessivamente molto positiva.

Nicolas Bourbaki è un personaggio immaginario (il cognome è quello di un generale francese, Charles Denis Bourbaki) , mentre è stata costituita nel 1935 una *Association des collaborateurs de Nicolas Bourbaki* che dispone di un ufficio presso la *École Normale Supérieure* in Parigi. Suoi membri fondatori sono Henri Cartan, Claude Chevalley, Jean Coulomb, Jean Delsarte, Jean Dieudonné, Charles Ehresmann, René de Possel, Szolem Mandelbrojt e André Weil. Le sue attività principali sono state la redazione degli *Éléments de Mathématique* e l'organizzazione di un *Séminaire Bourbaki*.

Il tentativo di classificare gli aspetti principali della matematica in alcune teorie fondamentali risponde all'esigenza di un crescente livello di astrazione, ma spesso questo porta a dimenticare l'aspetto "empirico" di questa scienza, necessario specialmente nella fase di apprendimento.

Un esempio di come la matematica risulti essere una "traduzione" ad un livello più astratto di concetti concreti può essere fornito dalla nozione di *numero*. Il numero nasce dalla necessità di avere un controllo sull'ordine e sulla quantità degli oggetti (e.g. si pensi alle prime forme di commercio ed al problema di quantificare il valore degli oggetti da barattare). Il concetto di numero nasce da un'esperienza concreta e si sviluppa in astratto seguendo l'esigenza di eseguire dei calcoli. Il concetto di calcolo deve poi essere considerato in astratto e quindi separato dal particolare procedimento risolutivo – *algoritmo* – che si adopera per eseguirlo.

Esempio 0.0.1 (Somma)**.** Se si volesse eseguire la somma $3 + 5$ si potrebbe contare scorrendo in sequenza 3 dita di una mano e 5 dell'altra; ma per sommare $19 + 114$ si ha bisogno di un algoritmo e.g. la somma in colonna imparata alle scuole elementari.

In generale, si può procedere scomponendo il problema in sottoproblemi più semplici (e.g. $7253 + 3128 = 7253 + 8 + 20 + 100 + 3000$). Tuttavia, sono molti i calcoli che non si compiono facilmente pur avendo a disposizione un algoritmo

(e.g. l'estrazione della radice quadrata di un numero). ☒

L'uomo e il calcolatore

Come l'uso delle cifre indiane[1] ha permesso di abbandonare l'abaco (e conseguentemente ha richiesto lo sviluppo di nuovi algoritmi), così il moderno calcolatore (che fornisce un valido ausilio nel calcolo) forza alla ricerca di nuovi algoritmi adatti alla sua struttura.

Un calcolatore opera in maniera sequenziale ed ha poca capacità di sintesi; tuttavia esso ha un elemento fondamentale che gli permette di immagazzinare una grande quantità di dati: la *memoria*.

Al contrario, l'uomo fa uso della propria capacità *sincronica*. Questa gli permette, una volta acquisita una visione globale dei dati, di farne una sintesi.

Il calcolatore opera in binario i.e. mediante sequenze di bit; dunque si ha la necessità di opportuni algoritmi che, utilizzando soltanto questa base, portino la macchina a simulare il nostro modo di pensare. Considerando però la difficoltà dell'uomo nell'utilizzare il linguaggio binario, vengono impiegati varî *metalinguaggi* per comunicare con la macchina ad un livello più elevato (e quindi più vicino al nostro). Si è così sviluppata la *teoria dei linguaggi artificiali*.

[1] Quelle che comunemente sono chiamate "cifre arabe" sono, più specificamente, le cifre *gobar* importate dagli arabi.

Parte I

Aspetti introduttivi

Capitolo 1

Semantica dei linguaggi di programmazione

Un linguaggio di programmazione viene tipicamente utilizzato per esprimere processi di calcolo eseguibili da un calcolatore.

Il problema di fornire una *semantica* per un linguaggio di programmazione consiste nello specificare senza ambiguità quale sia il significato di un programma scritto nel linguaggio stesso.

I vantaggi di una definizione formale della semantica di un linguaggio di programmazione sono molteplici:

- l'*utente* del linguaggio viene messo in grado di comprendere con precisione il suo strumento di lavoro
- *chi deve implementare il linguaggio* su uno specifico sistema trae benefici in quanto essa rappresenta una specifica
- consente la *verifica formale di proprietà*
- permette di stabilire la correttezza dei compilatori, sia in fase di traduzione che di ottimizzazione, mediante lo studio dell'*equivalenza di programmi*[1]

Si può rivedere tutto il discorso da un altro punto di vista mediante la seguente osservazione. Un linguaggio è "in prima battuta" una sintassi, e spesso dà luogo ad un numero infinito di termini. Inoltre, governare una sintassi è tipicamente difficile. Quindi, si applica il seguente schema:

[1] Ad esempio, si può dire che due programmi sono equivalenti se producono lo stesso risultato, oppure quando è possibile sostituire l'uno con l'altro in ogni contesto senza produrre effetti collaterali "visibili".

$$Sintassi \overset{interpr}{\longrightarrow} Semantica$$

i.e. si associa una semantica con l'obiettivo di semplificare lo studio; in ogni caso avere a disposizione più strumenti per operare è indubbiamente un vantaggio.

Si cerca di stabilire una corrispondenza tra sintassi e semantica i.e. poter asserire che tutto ciò che è dimostrabile in sintassi è verificabile in semantica (praticamente si cercano le proprietà di correttezza e completezza). La correttezza sarà un requisito necessario, invece è difficile avere una sintassi completa.

Infine, si osservi che per un linguaggio si possono avere più semantiche (e.g. data la sintassi S e le semantiche S_1 e S_2 si può avere $S \to S_1, S \to S_2$) e una può fare da sintassi per l'altra.

Si presentano ora vari approcci per definire la semantica di un linguaggio di programmazione.

1.1 Semantica statica

Nella *semantica statica* i programmi vengono trattati come oggetti matematici; avendo questi un tipo, la semantica formale assegna un tipo a ciascun termine del linguaggio.

Il *sistema di tipi* di un linguaggio di programmazione (i.e. l'insieme delle regole che consentono di associare un tipo ai termini) rappresenta la *semantica statica*. Lo scopo principale di un sistema di tipi è quello di prevenire il verificarsi di errori durante l'esecuzione di un programma, ad esempio l'applicazione di funzioni a parametri scorretti.

1.2 Semantica dinamica

Nella *semantica dinamica* il "significato" di un programma può essere inteso come il suo comportamento durante l'esecuzione. La semantica può essere descritta in maniera informale (come nei manuali di programmazione) o formalmente, cioè applicando le conoscenze della matematica e della logica. Vi sono diversi approcci: *operazionale*, *assiomatico* e *denotazionale*.

1.2.1 Semantica operazionale

Nella *semantica operazionale* il "significato" di un programma viene descritto da una relazione matematica sui termini del linguaggio che induce una nozione

di valutazione. Praticamente, si individuano dei passi di computazione che possono essere interpretati durante l'esecuzione.

Una forma "primitiva" di semantica operazionale può consistere nel definire una macchina in grado di eseguire le frasi del linguaggio (e.g. una Macchina di Turing). Il punto debole di questo approccio consiste nel fatto che il significato di un programma dipende da qualcosa di "esterno" al linguaggio i.e. la computazione di una specifica macchina.

Basandosi sulla struttura sintattica dei programmi si definisce la *relazione di valutazione*. Si consideri il seguente esempio.

Esempio 1.2.1 (Valutazione)**.** Sia $Env \times Term$ l'insieme delle *configurazioni*, dove $Env = Var \xrightarrow{fin} Const$ è l'insieme degli *ambienti* che associano costanti a variabili, e $Term$ è l'insieme dei termini del linguaggio. La relazione di valutazione $\rightarrow \subseteq (Env \times Term) \times (Env \times Term)$ tra configurazioni può essere definita da regole del tipo:

$$(E_1, e_1) \rightarrow (E_2, e_2)$$

Ad esempio, il significato dell'espressione: $(4 + x) - 3$ nell'ambiente $\{(x, 2)\}$ può essere la computazione:

$$\begin{array}{ll} & \{(x,2)\}, (4+x)-3 \\ \rightarrow & \{(x,2)\}, (4+2)-3 \\ \rightarrow & \{(x,2)\}, (6)-3 \\ \rightarrow & \{(x,2)\}, 3 \end{array}$$

⊠

Il precedente tipo di valutazione viene chiamato "a piccoli passi" poiché rappresenta passi elementari di computazione. Per valutazione "a grandi passi" si intende la relazione di valutazione che associa termini a risultati[2], e tipicamente si denota con $\rightarrow^*$.

1.2.2 Semantica assiomatica

Nella *semantica assiomatica* il "significato" di un programma è determinato (nel contesto di una teoria assiomatica) dall'insieme delle proposizioni vere per il programma stesso. Un esempio di tale approccio è la *Logica di Hoare*. Per quanto riguarda la notazione,

$$\{P\}C\{Q\}$$

[2]La valutazione "a grandi passi" è la chiusura riflessiva e transitiva della valutazione "a piccoli passi".

Tabella 1.1: Charles Antony Richard Hoare

Tratto da [WiE06].
Sir *Charles Antony Richard Hoare* (Tony Hoare o C.A.R. Hoare, nato l'11 gennaio 1934), informatico inglese. Ideatore, nel 1960, dell'algoritmo di ordinamento *Quicksort*. Ha inoltre sviluppato la *Logica di Hoare*, e il linguaggio formale *Communicating Sequential Processes* (CSP) utilizzato nella specifica delle interazioni tra processi concorrenti. Nel 1980 ha ricevuto l'*ACM Turing Award* per "i suoi fondamentali contributi nel campo della definizione e progettazione dei linguaggi di programmazione".

denota il seguente scenario: siano P e Q proposizioni logiche, C un comando, se il comando C – eseguito in uno stato che soddisfa la proprietà P – termina, allora lo stato che si raggiunge soddisfa la proprietà Q (e.g. $\{x = 0\}\ x := x + 1\ \{x > 0\}$).

1.2.3 Semantica denotazionale

In questo corso si studierà più da vicino l'approccio relativo alla *semantica denotazionale*.

Nella semantica denotazionale, il "significato" di un programma viene assegnato interpretando i termini del linguaggio in oggetti matematici (numeri, elementi di strutture algebriche, ...). In questo modo si raggiunge uno dei principali obiettivi della semantica denotazionale: si specificano i costrutti dei linguaggi di programmazione nella maniera più astratta e indipendente dall'implementazione possibile; in questo modo si può avere una maggiore comprensione degli aspetti concernenti il linguaggio preso in considerazione. Infine, è importante verificare che la specifica denotazionale di un linguaggio di programmazione sia implementabile i.e. collegare la semantica denotazionale alla semantica operazionale.

Esempio 1.2.2 (Numeri naturali)**.** Un esempio di tale approccio può essere la rappresentazione dei numeri naturali formulata da Von Neumann: i numeri $0, 1, 2, \ldots$ vengono rappresentati da $\emptyset, \{\emptyset\}, \{\{\emptyset\}, \emptyset\}, \ldots$ ⊠

La semantica denotazionale è molto importante anche perché i linguaggi funzionali si traducono opportunamente in questa. Infatti, la semantica denotazionale nacque per dare supporto insiemistico al linguaggio funzionale λ-calcolo.

Tabella 1.2: John von Neumann

Tratto da [Wik06].

John von Neumann (Budapest, Ungheria, 28 dicembre 1903 - Washington, USA, 8 febbraio 1957) matematico ungaro-statunitense, una delle personalità scientifiche preminenti del XX secolo cui si devono fondamentali contributi in campi come teoria degli insiemi, fisica quantistica, economia, informatica, teoria dei giochi, fluidodinamica e in molti altri settori della matematica.

Il nome originale è Neumann János Lajos Margittai e cresce in una famiglia ebrea non praticante. Nel 1911 entra nell'*Evangelikus Gimnazium di Budapest*. Nel 1913 il padre acquisisce un titolo nobiliare e da allora si chiamerà von Neumann. Nel 1926 ottiene un PhD in matematica all'*Università di Budapest*, mentre contemporaneamente studia chimica in Svizzera.

Tra il 1926 e il 1930 è Privatdozent a Berlino. In questo periodo dà importanti contributi alla logica matematica, alla teoria degli insiemi e ai fondamenti matematici della meccanica quantistica.

Nel 1930 viene invitato alla *Princeton University* ed è uno dei primi 4 docenti scelti per l'*Institute for Advanced Study*. Di questa prestigiosa istituzione è professore dal 1933 fino alla morte.

Nel 1938 riceve il *Bochner Memorial Award* per i suoi lavori in analisi matematica.

Allo scoppio della II guerra mondiale cambia completamente i suoi interessi e si rivolge alla matematica applicata, alla modellizzazione e alla soluzione effettiva di problemi di grande importanza pratica. In particolare si occupa di teoria dei giochi e con *Oskar Morgenstern* nel 1944 pubblica il testo fondamentale *Theory of Games and Economic Behavior*.

Lavora poi al *Progetto Manhattan* per la costruzione delle prime bombe atomiche, a *Los Alamos* nella divisione incaricata della teoria e dei calcoli, in stretto contatto con *Hans Bethe* e *Victor Weisskopf*.

A von Neumann si deve la definizione del concetto di *MAD*, Mutually Assured Destruction, nozione che ha influenzato la strategia degli USA durante tutto il periodo della guerra fredda, e i cui principali sviluppatori nella teoria dei giochi, *Aumann* e *Schelling*, hanno vinto il Premio Nobel per l'economia nel 2005.

Come si vedrà in seguito, gli oggetti matematici che soggiacciono alla semantica denotazionale sono insiemi con struttura (infatti, in un linguaggio funzionale i termini del liguaggio sono funzioni e le funzioni sono insiemi).

Caratteristiche della semantica denotazionale Si presentano ora (a livello introduttivo) alcune caratteristiche della semantica denotazionale.

Innanzitutto, ad ogni termine P si può associare una denotazione $[\![P]\!]$, un oggetto matematico che rappresenta il contributo di P in ogni programma in cui occorre.

Inoltre, la denotazione di una frase è determinata dalla denotazione delle sue sottofrasi (per questo si dice che la semantica è *composizionale*).

Il seguente esempio mostra un esempio di composizionalità.

Esempio 1.2.3 (Composizionalità - if)**.** Siano le funzioni parziali:

$$[\![C]\!], [\![C']\!] : State \nrightarrow State$$

e la funzione:

$$[\![B]\!] : State \rightarrow \{true, false\}$$

si può definire:

$$[\![\texttt{if } B \texttt{ then } C \texttt{ else } C']\!] = \lambda s \in State.if\left([\![B]\!](s), [\![C]\!](s), [\![C']\!](s)\right)$$

dove:

$$if\,(b, x, x') = \begin{cases} x & \text{se } b = true \\ x' & \text{se } b = false \end{cases}$$

⊠

Si presenta ora un ulteriore esempio di composizionalità.

Esempio 1.2.4 (Composizionalità)**.** Sia $\rho = \{(x, 2)\}$ l'ambiente che associa alla variabile x il valore 2. Si ha:

$$[\![(4 + x) - 3]\!]_\rho = [\![4 + x]\!]_\rho - [\![3]\!]_\rho = 3$$

dove $[\![t]\!]_\rho$ denota l'interpretazione del termine t sotto l'ambiente ρ. ⊠

Approfondiremo in seguito (cap. 10) la composizionalità.

Definizione 1.2.5 (Composizione sequenziale)**.** *Dati due comandi C e C', si definisce la denotazione della* composizione sequenziale *dei due comandi $C; C'$ come:*

$$[\![C; C']\!] = [\![C']\!] \circ [\![C]\!] = \lambda s \in State.[\![C']\!]([\![C]\!](s))$$

data dalla composizione delle funzioni parziali $[\![C]\!], [\![C']\!] : State \nrightarrow State$ che sono le denotazioni dei comandi. ◊

Osservazione 1.2.6 (Composizione sequenziale in semantica operazionale). Si confronti l'approccio appena visto con l'approccio relativo alla semantica operazionale, dove per la composizione sequenziale si ha:

$$\frac{C, s \Downarrow s' \quad C', s' \Downarrow s''}{C; C', s \Downarrow s''}$$

$\Diamond$

Capitolo 2

I numeri naturali

I numeri naturali rappresentano un punto fondamentale di riferimento per quanto riguarda gli aspetti algebrici e d'ordine per una struttura matematica, in particolare nel caso in cui si sia interessati a fornire una semantica per i linguaggi di programmazione. Per avere una pproccio intuitivo a questa problematica in questo capitolo si presenterà dapprima l'assiomatizzazione dei numeri naturali dovuta a *Campano da Novara*, successivamente si tratterà il *principio di induzione* ed infine si introdurrà il concetto di *ordinamento*.

2.1 L'assiomatizzazione di Campano da Novara

La prima assiomatizzazione dei numeri naturali è dovuta a Campano da Novara, il quale formulò la seguente definizione.

Definizione 2.1.1 (Serie naturale dei numeri)**.** *Si dice* serie naturale dei numeri *quella per la quale il calcolo degli stessi avviene aggiungendo un'unità.* ◊

A questa definizione egli aggiungeva i seguenti assiomi (*petitiones*).

Assiomi 2.1.2 (Numeri naturali)**.**

1. *Di ogni numero si possono prendere quante copie o quanti multipli si vuole.*
2. *La serie dei numeri può procedere all'infinito.*
3. *Nessun numero può essere diminuito all'infinito.*

◊

Tabella 2.1: Campano da Novara

Tratto da [Gal06].
Giovanni Campano da Novara, Novara 1220/30ca. - Viterbo 1296.
Tra i più importanti scienziati e matematici del secolo XIII, già nel 1255 aveva composto un importante commento agli *Elementa di Euclide*, introducendo un sistema di calcolo degli angoli del pentagono. Come cappellano papale è documentato alla Curia pontificia verso il 1261 ed è considerato in questo momento uno dei quattro migliori matematici del suo tempo. Su ordine di *Urbano IV* (1261-64) realizzerà la *Theorica Planetarum*. Dopo trent'anni di presenza nella curia pontificia a contatto con i maggiori filosofi naturali dell'epoca, raccolse un enorme patrimonio immobiliare, stimato alla morte da un ambasciatore aragonese in più di 12 mila fiorini: una ricchezza legata con ogni probabilità alla sua attività di medico. Negli ambienti curiali fu assai fortunata una benefica pillola da lui fabbricata, di cui poi si lesse la ricetta nel *Breviarium Praticae*. Si ricorda anche una sua splendida dimora presso Viterbo, in una zona di bagni termali, nella quale abitò negli ultimi anni della sua vita. Di lui ci restano l'*Abbreviatio equatorii planetarum*, la *Theorica planetarum*, del *Canon pro minutionibus et purgationibus*, il *Computus maior*, del *Tractatus de sphera*, i commenti ad *Euclide* e all'*Almagesto*. Secondo una recente ipotesi sarebbe a lui attribuibile anche lo *Speculum astronomiae*, importantissimo catalogo di opere astrologiche, che distingueva magia lecita dall'illecita.

Si osservi che, per quanto riguarda l'assioma 1, il numero è trattato come un termine del linguaggio (e quindi può essere ripetuto quante volte si vuole).

In questo modo egli fornì una prima forma del *principio di induzione* (a volte abbreviato con IP). Questa assiomatizzazione, pur non caratterizzando completamente i numeri naturali (indicati con $\mathbb{N}$), ne caratterizza comunque un procedimento costruttivo di generazione. Naturalmente manca la richiesta (per quell'epoca inconcepibile) che il punto di partenza sia unico.

2.2 Principio di induzione

Il principio di induzione può essere enunciato come segue.

Proposizione 2.2.1 (Principio di induzione)**.** *Sia P una proprietà su $\mathbb{N}$. Se valgono:*

- $P(0)$
- $P(n) \Rightarrow P(s(n))$[1]

allora P è vera per $\mathbb{N}$. ◊

Dimostrazione. Si procederà utilizzando l'intuizione di Campano da Novara.

Si supponga - per assurdo - che esista $m \in \mathbb{N}$ tale che $P(m)$ non è vera. Quindi, denotando con $p(m)$ il predecessore di m, si avrà che $P(p(m))$ non è vera (altrimenti si avrebbe che $P(m) = P(s(p(m)))$ è vera).

Analogamente, anche $P(p(p(m)))$ non è vera. Si prosegue in questo modo; si osservi che non si può continuare all'infinito (per l'assioma 3 di 2.1.2). Quindi, ammessa l'esistenza di un primo numero 0, si arriverebbe all'assurdo che $P(0)$ è falsa (contro l'ipotesi). □

Il principio di induzione può essere equivalentemente espresso nel modo seguente.

Definizione 2.2.2 (IP - formulazione equivalente)**.** *Sia $A \subseteq \mathbb{N}$ tale che*

- $0 \in A$
- $\forall n \in A \;\; s(n) \in A$

allora $A = \mathbb{N}$. ◊

[1] $s(n)$ denota il successore di n.

Si osservi che la seconda ipotesi su A precisa che questo è chiuso rispetto alla funzione successore.

Si definiscono ora alcune funzioni su $\mathbb{N}$.

Definizione 2.2.3 (Funzioni su $\mathbb{N}$)**.** *Si possono definire su* $\mathbb{N}$ *le tre funzioni* s*,* 0 *e* $+$ *:*

1. *una funzione iniettiva* successore*:*

$$\begin{array}{rcl} s : \mathbb{N} & \to & \mathbb{N} \\ n & \mapsto & s\,(n) \end{array}$$

2. *una costante* $0 \in \mathbb{N}$ *tale che* $\forall n \in \mathbb{N}\ \ 0 \neq s(n)$

3. *una funzione* somma*:*

$$\begin{array}{rcl} + : \mathbb{N} \times \mathbb{N} & \to & \mathbb{N} \\ (m, n) & \mapsto & m + n \end{array}$$

$\Diamond$

Il principio di induzione permette di dimostrare la validità di una proprietà o costruire una definizione di un insieme infinito di elementi gestendo un numero finito di casi. Seguono alcuni esempi di utilizzo del principio di induzione.

Esempio 2.2.4 (Somma)**.** Tramite i punti 1 e 2 della proposizione 2.2.3 si può definire la somma per induzione.

Si consideri la somma, definita come funzione di due variabili m ed n. Essendo il dominio di tale funzione il prodotto cartesiano $\mathbb{N} \times \mathbb{N}$ si possono far variare indipendentemente gli elementi che lo costituiscono i.e. si può pensare $+$ come una funzione sulla variabile n fissando m come parametro; poi viceversa. Dunque, fissato il primo termine m, si definisce la seguente funzione:

$$\begin{array}{rcl} m + \_ : \mathbb{N} & \to & \mathbb{N} \\ 0 :\ m + 0 & \mapsto & m \\ s(h) :\ m + s(h) & \mapsto & s(m + h) \end{array}$$

Ora, avendo definito la funzione sulla variabile n, si pensa quest'ultima come un parametro e si ripete lo stesso ragionamento su m. $\boxtimes$

Esempio 2.2.5 (Successore)**.** Dimostriamo che il successore di m è $m + 1$:

$(m = 0)$

$s(m) = s(0) = 1 = 0 + 1 = m + 1$

$(m > 0)$

$m > 0 \Rightarrow m = s(m')$

$s(m) = s(s(m')) \stackrel{IH}{=} s(m' + 1) = s(m') + 1 = m + 1$ ⊠

Esercizio 2.2.6. Dimostrare che la somma verifica le seguenti proprietà: associatività, commutatività, esistenza dell'elemento neutro. ⊠

2.3 Ordinamento

Si introduce in $\mathbb{N}$ l'*ordinamento* $\leq$.

Definizione 2.3.1 ($\leq$)**.** $n \leq m \stackrel{def}{\Leftrightarrow} \exists h \in \mathbb{N}$ *tale che* $n + h = m$. ◊

Definizione 2.3.2 (Buon ordinamento)**.** *L'ordinamento* $\preceq$ *su* A *è un* buon ordinamento *se* $\forall S \subseteq A$ *tale che* $S \neq \emptyset$*,* S *ha un primo elemento.* ◊

Proprietà 2.3.3 ($\leq$ è un buon ordinamento)**.** *L'*ordinamento $\leq$ *su* $\mathbb{N}$ *è un* buon ordinamento*.* ◊

Per verificare questa proprietà informalmente, comunque si prendano due elementi in $\mathbb{N}$, si considera l'insieme S formato solo da essi e si stabilisce quale elemento sia il primo.

Si tratta di una proprietà del secondo ordine (i.e. si quantifica sugli insiemi), equivalente al principio di induzione.

Proprietà 2.3.4 (Equivalenza IP - buon ordinamento)**.** *Il principio di induzione è equivalente al principio del buon ordinamento.* ◊

Dimostrazione. Qui si mostra solo che il principio del buon ordinamento implica il principio di induzione.

Si considerano il principio del buon ordinamento e le premesse del principio di induzione i.e. $P(0)$ e $P(n) \Rightarrow P(n+1)$. La tesi è $\forall n \in \mathbb{N}\ \ P(n)$.

Sia $S = \{n \in \mathbb{N} \mid \neg P(n)\}$. Per il principio del buon ordinamento esiste un elemento $m \in S$ minimo; $m \neq 0$ poiché $P(0)$ per ipotesi.

Quindi, $m - 1 \in \mathbb{N}$ i.e. $m - 1 \notin S$ (poiché m è il minimo elemento di S). Allora, $P(m-1)$, e $P(m-1) \Rightarrow P(m)$ per ipotesi; questa è una contraddizione.

Quindi, S non può avere un elemento minimo, per il principio del buon ordinamento segue che $S = \emptyset$. Allora, $\forall n \in \mathbb{N}\ \ P(n)$. □

Osservazione 2.3.5 (Ordinamento totale). Se un ordinamento è un buon ordinamento allora questo è *totale*[2]. ◊

[2]Un insieme M si dice *totalmente ordinato* sse è ordinato e $\forall m, n \in M$ $\ m$ e n sono confrontabili.

Dimostrazione. Si considerino tutte le coppie. □

Osservazione 2.3.6. Ogni insieme bene ordinato ammette il principio di induzione. ◊

Esempio 2.3.7 (Buon ordinamento)**.**

- Sia $\mathbb{Z}'$ l'insieme dei numeri interi riordinato come:

  $$0, 1, 2, 3, 4, \ldots, -1, -2, -3, -4, \ldots$$

  $\mathbb{Z}'$ è un buon ordinamento rispetto a $\leq$.

- Esempi di insiemi non bene ordinati sono $\mathbb{Q}$ e $\mathbb{Z}$ con l'ordinamento consueto (si consideri e.g. $\{x \mid x > 1\}$, si può verificare che non esiste un primo elemento nei due insiemi considerati).

⊠

Su un insieme come quello appena visto (e.g. $\mathbb{Z}'$) non è applicabile il principio di induzione nella forma vista in precedenza (poiché non è definita la funzione s). Si rende quindi necessaria la seguente formulazione alternativa (dimostrabile equivalente).

Definizione 2.3.8 (Principio di induzione forte)**.** *Sia $A \subseteq \mathbb{N}$ tale che:*

- $0 \in A$ *e*
- $\downarrow m \subseteq A \Rightarrow m \in A$[3]

allora $A = \mathbb{N}$. ◊

La seconda ipotesi indica che se tutti gli elementi che precedono m sono in A, allora anche m è in A.

Esempio 2.3.9 (Principio di induzione forte)**.** Questa formulazione dell'IP viene utilizzata nella dimostrazione della finitezza della costruzione dei tableau semantici in [Ben98], dove, sommariamente, si assegna ad ogni formula logica una complessità $\mathcal{C}(F)$ dimostrando che per ogni passo questo valore diminuisce. ⊠

Esercizio 2.3.10 (Proprietà di $\leq$)**.** Verificare che:

- $\leq$ in $\mathbb{N}$ è una relazione d'ordine [4]

[3] $\downarrow m$ denota il *segmento iniziale* i.e. l'insieme degli elementi che precedono m.

[4] Una relazione si dice d'*ordine* (o d'ordine parziale) se è riflessiva, antisimmetrica e transitiva.

- $\forall n \in \mathbb{N}^+ \;\; 0 < n$ [5]
- $\forall n \in \mathbb{N} \;\; n < s(n)$

$\boxtimes$

Si osservi che tutte le proprietà dei numeri naturali si possono ottenere a partire dagli assiomi definiti da Giuseppe Peano.

Si richiamano ora altre forme di principio di induzione, presentate in maniera informale.

Una prima forma vale per termini che hanno una struttura "complessa" e.g. termini costruiti a partire da una grammatica. Quindi, il *principio di induzione strutturale* considera un insieme T di termini (di struttura complessa), si mappa $T \xrightarrow{\mathcal{C}} \mathbb{N}$ dove per ogni $t \in T$ è definita la complessità $\mathcal{C}(t)$, e poi si fa induzione su $\mathbb{N}$.

Un'altra forma si ha quando non si ha un unica variabile su cui fare induzione: si ha il *principio di induzione multipla* (e.g. doppia, tripla, ...). Consideriamo ad esempio l'induzione doppia. Si mappa $\mathbb{N} \times \mathbb{N} \to \mathbb{N}$, e poi si fa induzione su $\mathbb{N}$.

Esempio 2.3.11 (Induzione doppia)**.** Consideriamo il prodotto fra due numeri naturali, $n \cdot m$. Si fissa una variabile e si fa induzione rispetto all'altra i.e.

- $n \cdot 0 = 0$
- $n \cdot s(m) = (n \cdot m) + n$

$\boxtimes$

Esercizio 2.3.12 (Monotonia somma)**.** Dimostrare che la somma è monotona rispetto all'ordine i.e.

$$\forall n, n', m \in \mathbb{N} \;\; n \leq n' \Rightarrow n + m \leq n' + m$$

$\boxtimes$

Definizione 2.3.13 (Ordinamento lessicografico)**.** *Siano* $a, b, c, d \in \mathbb{N}$.

$$(a, b) \leq (c, d) \overset{def}{\Leftrightarrow} a < c \;\vee\; (a = c \;\wedge\; b \leq d)$$

$\lozenge$

Un esempio di tale ordinamento è quello utilizzato nei vocabolari.

[5] $\mathbb{N}^+ = \mathbb{N} \setminus \{0\}$.

Tabella 2.2: Giuseppe Peano

Tratto da [Gal06].
Giuseppe Peano, Cuneo 1858 - Torino 1932.
Assistente all'*Università di Torino* di Angelo Genocchi, divenne professore di calcolo infinitesimale presso la stesso ateneo nel 1890. Precisò la nozione di *limite superiore* e fornì il primo esempio di una curva che "riempie" un'area. Contribuì a fondare la teoria del *calcolo vettoriale* e realizzò importanti ricerche sulla teoria delle *equazioni differenziali ordinarie*, ideando un metodo di integrazione per successive approssimazioni. Come logico dette un eccezionale contributo alla *logica delle classi*, elaborando un simbolismo di grande chiarezza e semplicità. Coltivò anche l'ideale leibniziano della costruzione di una lingua universale (*latino sine flexione*). Ebbe ampi riconoscimenti negli ambienti filosofici più aperti alle esigenze e alle implicazioni critiche della nuova logica formale.

Osservazione 2.3.14 (Proprietà ordinamento lessicografico). L'ordinamento lessicografico è un buon ordinamento. Infatti, sia $K \subseteq \mathbb{N} \times \mathbb{N}$. Bisogna provare che K ha un minimo. Siano $(n_0, m_0), (n_1, m_1) \in K$. Se $n_0 \neq n_1$ allora si confrontano n_0 e n_1, altrimenti m_0 e m_1.

Quindi, l'ordinamento appena introdotto risulta essere un ordinamento totale. ◊

In $\mathbb{N} \times \mathbb{N}$ si può definire un altro ordinamento, chiamato *ordinamento prodotto*.

Definizione 2.3.15 (Ordinamento prodotto)**.** *Siano* $n, m, n', m' \in \mathbb{N}$.

$$(n, m) \leq (n', m') \overset{def}{\Leftrightarrow} n \leq n' \wedge m \leq m'$$

◊

L'ordinamento appena introdotto risulta essere *parziale*. Questo può essere molto utile - come vedremo in seguito - per il fatto che opera indipendentemente sugli elementi.

2.4 Insiemi Ordinati

Definizione 2.4.1 (Insieme parzialmente ordinato)**.** *Un* insieme parzialmente ordinato $(D, \leq)$ *è un insieme* D *su cui è definita una relazione binaria* $\leq$ *tale che* $\forall x, y, z \in D$ *si ha:*

1. $x \leq x$ *(proprietà riflessiva)*

2. $x \leq y \wedge y \leq z \Rightarrow x \leq z$ *(proprietà transitiva)*

3. $x \leq y \wedge y \leq x \Rightarrow x = y$ *(proprietà antisimmetrica)*

$\Diamond$

A livello di terminologia, spesso si utilizza il nome *poset* per indicare un insieme parzialmente ordinato, l'insieme D è chiamato *dominio sottostante* del poset, e la relazione $\leq$ è chiamata *relazione d'ordine*.

Definizione 2.4.2 (Monotonia)**.** *Una funzione $f : D \to E$ tra insiemi parzialmente ordinati è* monotona *se e solo se:*

$$\forall d, d' \in D \quad d \sqsubseteq d' \Rightarrow f(d) \sqsubseteq f(d')$$

ovvero conserva l'ordine rispetto alla relazione $\sqsubseteq$. $\Diamond$

Definizione 2.4.3 (Minimo e massimo)**.** *Supponiamo che $(D, \leq)$ sia un insieme parzialmente ordinato e sia $S \subseteq D$. Un elemento $d \in S$ è l'elemento* minimo *di S se soddisfa la seguente condizione:*

$$\forall x \in S \;\; d \leq x$$

Il massimo *è definito dualmente.* $\Diamond$

Definizione 2.4.4 (Massimo comune minorante)**.** *Sia (a, b) una coppia di elementi in un insieme parzialmente ordinato $(X, \leq)$. Si definisce* massimo comune minorante *$a \cap b$ l'elemento tale che:*

- $a \cap b \leq a \;\wedge\; a \cap b \leq b$
- $\forall c \in X \;\; c \leq a \;\wedge\; c \leq b \;\; \Rightarrow \;\; c \leq a \cap b$

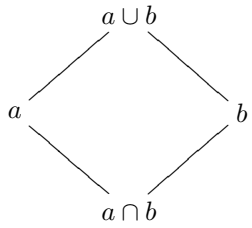
$\Diamond$

Definizione 2.4.5 (Minimo comune maggiorante)**.** *Dualmente, sia (a, b) una coppia di elementi in un insieme parzialmente ordinato $(X, \leq)$. Si definisce* minimo comune maggiorante *$a \cup b$ l'elemento tale che:*

- $a \cup b \geq a \;\wedge\; a \cup b \geq b$
- $\forall c \in X \;\; c \geq a \;\wedge\; c \geq b \;\; \Rightarrow \;\; c \geq a \cup b$

$\Diamond$

Attraverso l'utilizzo dei diagrammi di Hasse le ultime due definizioni possono essere rappresentate nel seguente modo:



Esempio 2.4.6 (Minimo comune maggiorante)**.** Si consideri l'insieme $\mathbb{N}$ dei naturali e la relazione di divisibilità (che è d'ordine parziale): il massimo comune minorante coincide con il Massimo Comune Divisore ed il minimo comune maggiorante con il minimo comune multiplo. ⊠

Definizione 2.4.7 (Catena)**.** *Una* catena *in un insieme parzialmente ordinato* $(D, \sqsubseteq)$*, è una sequenza di elementi di* D $d_0, d_1, \ldots$ *tale che:*

$$d_0 \sqsubseteq d_1 \sqsubseteq d_2 \sqsubseteq d_3 \sqsubseteq ...$$

◊

Definizione 2.4.8 (Confine superiore di una catena)**.** *Un* confine superiore *di una catena è un elemento* $d \in D$ *tale che* $\forall n \in \mathbb{N}\ \ d_n \sqsubseteq d$*.* ◊

Definizione 2.4.9 (Minimo confine superiore)**.** *Qualora esista il confine superiore, il* minimo confine superiore*, (least upper bound – lub), di una catena* $d_0, d_1, \ldots$ *è:*

$$\bigsqcup_{n \geq 0} d_n$$

Si ha che:

$$\forall m \in \mathbb{N} \quad d_m \sqsubseteq \bigsqcup_{n \geq 0} d_n \tag{2.1}$$

$$\forall d \in D \quad (\forall m \geq 0\ d_m \sqsubseteq d) \quad \Rightarrow \quad \bigsqcup_{n \geq 0} d_n \sqsubseteq d \tag{2.2}$$

◊

Capitolo 3

Strutture algebriche

3.1 Reticolo algebrico

Definizione 3.1.1 (Reticolo)**.** *Un* reticolo $(X, \sqsubseteq, \cap, \cup)$ *è un insieme parzialmente ordinato tale che per ogni coppia di elementi* (a, b) *esiste un massimo comune minorante* $a \cap b$ *ed un minimo comune maggiorante* $a \cup b$*.* $\Diamond$

Esempio 3.1.2 (Reticolo)**.** Sia X un insieme, $(\mathcal{P}(X), \subseteq, \cap, \cup)$ è un reticolo. $\boxtimes$

Proprietà 3.1.3 (Proprietà di $\cap, \cup$)**.** *Sia* $(X, \sqsubseteq, \cap, \cup)$ *un reticolo, le operazioni rie* $\cap$ *e* $\cup$ *godono delle seguenti proprietà:*

1. $a \cap a = a, \quad a \cup a = a$ *(idempotenza)*
2. $a \cap b = b \cap a, \quad a \cup b = b \cup a$ *(commutatività)*
3. $(a \cap b) \cap c = a \cap (b \cap c), \quad (a \cup b) \cup c = a \cup (b \cup c)$ *(associatività)*
4. $a \cap (a \cup b) = a, \quad a \cup (a \cap b) = a$ *(assorbimento)*

$\Diamond$

Osservazione 3.1.4. Se $(X, \cap, \cup)$ è un insieme con due operazioni che godono delle proprietà 3.1.3, definendo su X la relazione d'ordine:

$$a \sqsubseteq b \;\Leftrightarrow\; a \cap b = a \;\wedge\; a \cup b = b$$

allora $\cap$ rappresenta il massimo comune minorante e $\cup$ il minimo comune maggiorante e conseguentemente si ottiene un reticolo. $\Diamond$

Definizione 3.1.5 (Reticolo distributivo)**.** *Un reticolo* $(X, \sqsubseteq, \cap, \cup)$ *si dice* distributivo *se per ogni* $a, b, c \in X$*:*

1. $a \cup (b \cap c) = (a \cup b) \cap (a \cup c)$

2. $a \cap (b \cup c) = (a \cap b) \cup (a \cap c)$

$\Diamond$

Definizione 3.1.6 (Reticolo complementato)**.** *Un reticolo* $(X, \sqsubseteq, \cap, \cup)$ *si dice complementato se possiede un minimo che indicheremo con 0, un massimo che indicheremo con 1 e se per ogni* $a \in X$ *esiste* $a' \in X$ *(detto* complemento *di a) tale che*

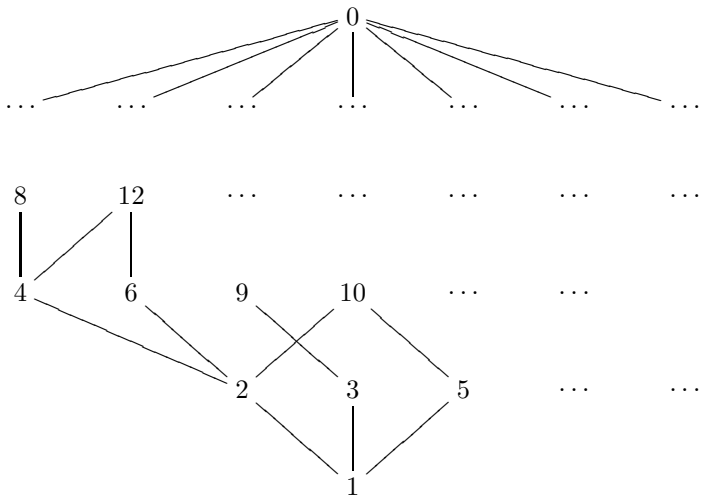
$$a \cap a' = 0 \text{ e } a \cup a' = 1$$

$\Diamond$

Definizione 3.1.7 (Reticolo completo)**.** *Un reticolo si dice* completo *se ogni sottoinsieme contiene il minimo comune maggiorante.* $\Diamond$

Esempio 3.1.8 (Reticolo completo)**.** Esempi di reticoli completi sono l'insieme dei numeri naturali $\mathbb{N}$ e l'insieme dei numeri reali positivi $\mathbb{R}^+$. $\boxtimes$

Esempio 3.1.9 (Reticolo non completo)**.** Si consideri il reticolo seguente, costruito in base alla relazione di divisibilità:



Questo reticolo ha un massimo in quanto lo 0 è in cima all'albero poiché è divisibile per tutti gli elementi sotto di esso. Ma questo reticolo non è completo, perché si può trovare un sottoinsieme che non contiene il minimo comune maggiorante. $\boxtimes$

3.2 Algebra di Boole

Definizione 3.2.1 (Algebra per $\mathcal{P}(X)$)**.** *Sia* $(\mathcal{P}(X), \cap, \cup, ()^C, \emptyset, X, \subseteq)$ *un'algebra per* $\mathcal{P}(X)$*, e sia la corrispondente algebra di Boole* $(\mathbb{B}, \cdot, +, ()', 0, 1, \subseteq)$*, con 0 elemento minimo e 1 elemento massimo nella relazione* $\subseteq$*. Si ha che* $+$ *è il minimo comune maggiorante e* $\cdot$ *è il massimo comune minorante.* ◊

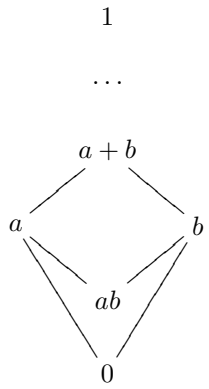
Definizione 3.2.2 (Algebra di Boole)**.** *Si definisce* algebra di Boole *un reticolo distributivo complementato.* ◊

Teorema 3.2.3 (Teorema di Stone)**.** *Ogni algebra di Boole è isomorfa:*

- *ad un'algebra di* $\mathcal{P}(X)$ *se è finita*
- *ad un sottoinsieme di un'algebra di* $\mathcal{P}(X)$ *se è infinita*

◊

Esempio 3.2.4. Si può "visualizzare" la struttura dell'algebra di Boole come segue. $a + b$ = minimo comune maggiorante di (a, b), ab = massimo comune minorante di (a, b).



⊠

Proprietà 3.2.5 (Algebra di Boole)**.**

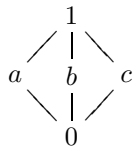
- *Commutatività:* $a + b = b + a$, $ab = ba$
- *Associatività:* $(a + b) + c = a + (b + c)$, $a(bc) = (ab)c$
- *Idempotenza:* $a + a = a$, $a \cdot a = a$
- *Distributività:*
  $a(b + c) = ab + ac$
  $a + (bc) = (a + b)(b + c)$
  $a + 0 = a$, $a \cdot 0 = 0$
  $a + 1 = 1$, $a \cdot 1 = a$

- *Complemento:* $a' \cdot a = 0$, $a' + a = 1$, $(a')' = a$

$\Diamond$

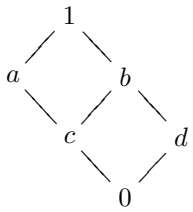
Esempio 3.2.6 (Come non è fatta un'algebra di Boole)**.**

- Un albero: ha massimo comune minorante ma non minimo comune maggiorante e la somma (poiché non ha cicli e quindi non si può "chiudere")
- Una struttura a "diamante", poiché impedisce al reticolo di essere distributivo



  Infatti ha massimo comune minorante e minimo comune maggiorante ma non è distributivo: $a(b + c) = ab + ac \Rightarrow a \cdot 1 = 0 + 0$, ma $a \neq 0$

- Un reticolo distributivo non complementato



  Per ogni elemento deve valere $a \cdot a' = 0$ e $a' + a = 1$, ma $b \cdot 0 = 0$, $b + 0 = 0$ e $c \cdot 0 = 0$, $c + 0 = 0$.

$\boxtimes$

Capitolo 4

Algebra dei linguaggi

4.1 Introduzione

La *teoria dei linguaggi*, nata inizialmente dallo studio dei linguaggi naturali, si è sviluppata sui linguaggi artificiali e sulla definizione delle regole che li caratterizzano.

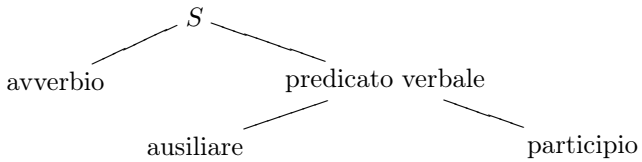
Tipicamente, non si è interessati a tutte le frasi, ma solo ad alcune particolari successioni di lessemi.

Le successioni per noi "buone" potrebbero ad esempio essere scelte tra quelle *sintatticamente corrette*, mentre un altro criterio di selezione potrebbe essere invece la richiesta che le parole siano *semanticamente significanti*.

Esempio 4.1.1. Si consideri la frase:

"Domani sono stato piovuto"

Indicando tale affermazione con la lettera S (dall'inglese *sentence*), possiamo schematizzarla nel seguente modo:



Nonostante la frase sia strutturalmente corretta, possiamo evidenziare degli errori nella sintassi: l'ausiliare è un participio passato del verbo 'essere' ed è passivo, mentre 'piovere' è un verbo intransitivo e richiede una forma attiva, l'avverbio 'domani' è futuro e richiede quindi la forma futura dell'ausiliare.

Invece, l'affermazione:

"Ieri ho mangiato"

nonostante abbia la stessa struttura della frase precedente, è sintatticamente corretta.

Si consideri ora la frase:

"Per sarà il certamente"

Questa non corrisponde a una struttura sintattica lecita in italiano.

La frase:

"il certamente è un avverbio"

è invece sintatticamente corretta poiché si è sostantivato l'avverbio, che ha quindi un ruolo diverso da quello abituale. ⊠

Si consideri ora la semantica. In generale è piuttosto difficile stabilire se una frase sia semanticamente corretta, in quanto il significato delle parole può cambiare a seconda del ruolo che si attribuisce loro. Basti pensare ai giochi linguistici in cui si avventurano i poeti.

4.2 Definizioni

Introduciamo ora alcune definizioni utili per il prosieguo.

Definizione 4.2.1 (Alfabeto, A^*, stringa vuota)**.** *Un* alfabeto A *è un insieme di simboli non vuoto e finito. Si denota con* A^* *l'insieme di tutte le stringhe di lunghezza finita di elementi in* A*. In particolare,* A^* *contiene anche la* stringa vuota*, denotata con* ε*.* ◊

Esempio 4.2.2. Sia $A = \{(,)\}$, A^* è l'insieme di tutte le stringhe finite di parentesi. ⊠

Definizione 4.2.3 (Linguaggio)**.** *Un linguaggio* L *sull'alfabeto* A *è un sottoinsieme di* A^* *i.e.* $L \in \mathcal{P}(A^*)$*.* ◊

Si ricordi che in $\mathcal{P}(A^*)$ sono definite le seguenti operazioni: intersezione ($\cap$), unione ($\cup$) e complemento ($\cdot'$).

Definizione 4.2.4 (Prodotto di Frobenius)**.**

$$\forall L, M \in \mathcal{P}(A^*) \quad L \cdot M \stackrel{def}{=} \{ww' \mid w \in L, w' \in M\}$$

◊

Tale prodotto gode delle seguenti proprietà.

Proprietà 4.2.5 (Prodotto di Frobenius)**.**

- Associatività*:* $(L \cdot M) \cdot N = L \cdot (M \cdot N)$
- *esiste un* elemento neutro $\{\varepsilon\}$ *tale che* $\{\varepsilon\} \cdot L = L \cdot \{\varepsilon\} = L$
- *esiste un elemento* $\emptyset$ annullatore *tale che* $\forall L \in \mathcal{P}(A^*)\ \ \emptyset \cdot L = \emptyset \wedge L \cdot \emptyset = \emptyset$
- Monotonia*:* $L \subseteq L'$ *e* $M \subseteq M' \Rightarrow L \cdot M \subseteq L' \cdot M'$

$\Diamond$

Si osservi che tale prodotto non è commutativo.

Definizione 4.2.6 (Derivata di un linguaggio)**.** *La* derivata sinistra *di* L *rispetto ad* M *è l'insieme:*

$$[L \setminus M] = \{w \mid L \cdot \{w\} \subseteq M\}$$

$\Diamond$

Esempio 4.2.7 (Derivata di un linguaggio)**.** Siano gli insiemi $M = \{ab, abc, b, bb\}$ ed $L = \{a\}$, allora $[L \backslash M] = \{b, bc\}$.

Cioè, la derivata sinistra di L rispetto ad M è il più grande insieme tale che $L \cdot [L \backslash M] = \{a\} \cdot \{b, bc\} = \{ab, abc\} \subseteq M$. ⊠

Esempio 4.2.8. Siano gli insiemi $M = \{ab, abc, b, bb\}$ e $L = \{a, b\}$, allora $[L \backslash M] = \{b\}$.

Infatti $L \cdot [L \backslash M] = \{a, b\} \cdot \{b\} = \{ab, bb\} \subseteq M$. ⊠

Proposizione 4.2.9.

$$[L \cdot N \subseteq M \Leftrightarrow N \subseteq L \backslash M]$$

$\Diamond$

Dimostrazione. Questo vale perché $[L \backslash M]$ è il più grande insieme tale che $L \cdot [L \backslash M] \subseteq M$. □

La proposizione precedente presenta un'analogia con il seguente terorema, utile (tra le altre cose) per dimostrare l'esistenza del complemento in una struttura.

Teorema 4.2.10 (Teorema Deduttivo)**.**

$$\frac{A \wedge B \vdash C}{B \vdash A \to C}$$

$\Diamond$

Capitolo 5

Monoidi

Definizione 5.0.11 (Monoide)**.** *Un* monoide $(M, +, u)$ *è un semigruppo* [1] *dotato di elemento neutro* u*, i.e. deve valere:*

$$\begin{array}{ll} M1)\ (x+y)+z = x+(y+z) & \forall x, y, z \in M \\ M2)\ x+u = u+x = x & \forall x \in M \end{array}$$

$\Diamond$

Definizione 5.0.12 (Monoide ordinato)**.** *Un monoide* $(M, +, u)$ *è* ordinato *se la somma è una funzione monotona rispetto a* $\leq$*, i.e.* $\forall n, n', m \in \mathbb{N}\ \ n \leq n' \Rightarrow n + m \leq n' + m$*.* $\Diamond$

Nel seguito, a volte si chiamerà un monoide $(M, +, u)$ semplicemente con M (qualora ciò non provochi ambiguità).

Esempio 5.0.13 (Monoidi)**.**

- $(\mathbb{N}, +, 0)$ è un monoide ordinato rispetto a $\leq$.
- L'insieme delle classi resto modulo n, $(\mathbb{Z}_n, +, [0])$, è un esempio di monoide non ordinato.

Infatti si ha la relazione:

$$\underbrace{[1] + [1] + ... + [1]}_{n} = [0]$$

Questa relazione esplicita un vincolo che, in particolare, impedisce l'ordinamento. Qui si è invece interessati a trattare *strutture libere* (cfr. def. 5.0.18). $\boxtimes$

[1] Insieme su cui è definita una operazione binaria associativa.

Definizione 5.0.14 (Omomorfismo di monoidi)**.** *Un* omomorfismo di monoidi *è una funzione*

$$f : (M, \circ, 1_M) \to (N, \bullet, 1_N)$$

tale che

$$\begin{cases} f(x \circ y) = f(x) \bullet f(y) \\ f(1_M) = 1_N \end{cases}$$

$\Diamond$

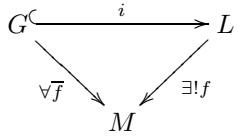
Definizione 5.0.15 (Insieme di generatori)**.** $G \subseteq M$ *è un* insieme di generatori *se ogni elemento di* M *è una combinazione finita di elementi di* G*.* $\Diamond$

Osservazione 5.0.16 (Insieme di generatori minimo). Ogni monoide M ammette sempre un insieme di generatori (almeno $G = M$); in realtà si è interessati all'insieme di generatori minimo. $\Diamond$

Esempio 5.0.17 (Insieme di generatori)**.** $(\mathbb{N}, +, 0)$ ammette l'insieme di generatori $\{1\}$. Infatti, ogni elemento $n \in \mathbb{N}$ si può esprimere come $\sum_{i=1}^{n} 1$ (l'elemento 0 si ottiene ponendo nella somma l'indice $i = 0$). $\boxtimes$

Definizione 5.0.18 (Monoide libero)**.** *Un monoide* L *generato da* G *si dice* libero *su* G *(si denota* $L(G)$ *o* G^**), se per ogni monoide* M *e per ogni* $\overline{f} : G \to M$*, esiste un unico morfismo* $f : L \to M$ *tale che* $f|_G = \overline{f}$*.*

Questa definizione si può alternativamente esprimere dicendo che esiste un unico morfismo tra L *ed* M *che rende commutativo il seguente diagramma:*



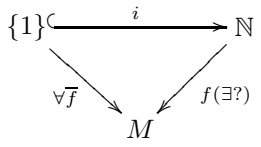
ovvero $\forall g \in G \ \ \overline{f}\,(g) = f\,(i\,(g))$*, dove* $i : G \subseteq L \to L$ *è l'inclusione.* $\Diamond$

Osservazione 5.0.19 (Spazio vettoriale). Si noti l'analogia con la proprietà che uno spazio vettoriale è libero sulla sua base. $\Diamond$

Esempio 5.0.20 (Monoide libero)**.** $\mathbb{N}$ è il monoide libero generato da $\{1\}$.

È già stato osservato che $\mathbb{N}$ è un monoide ed è generato da $\{1\}$ (es. 5.0.17). Verifichiamo ora che è libero.

Per ogni monoide $(M, \bullet, u)$ e $\forall \overline{f} : \{1\} \to M$, si vuole trovare un morfismo $f : \mathbb{N} \to M$ tale che $f|_{\{1\}} = \overline{f}$.



Affinché $f : \mathbb{N} \to M$ sia un morfismo, l'unica definizione possibile è:

$$f(n) = f(\underbrace{1 + \ldots + 1}_{n\ volte}) = \underbrace{f(1) \bullet \ldots \bullet f(1)}_{n\ volte}$$
$$f(1) = u$$

Infatti deve essere $f(1) = \overline{f}(1)$ affinché il diagramma sia commutativo. ⊠

Proposizione 5.0.21. *$\mathbb{N}$ è, a meno di isomorfismi, l'unico monoide libero generato da un elemento.* ◊

Dimostrazione. Sia $(\mathbb{N}', +', 1')$ un altro monoide libero generato dall'elemento $\{1'\}$.

Poiché $\mathbb{N}$ è libero, considerando $M = \mathbb{N}'$, si ha che esiste un solo omomorfismo α:

$$\begin{array}{rcl} \alpha : \mathbb{N} & \xrightarrow{!} & \mathbb{N}' \\ 1 & \mapsto & 1' \end{array}$$

Analogamente, essendo $\mathbb{N}'$ libero, esiste un solo omomorfismo α':

$$\begin{array}{rcl} \alpha' : \mathbb{N}' & \xrightarrow{!} & \mathbb{N} \\ 1' & \mapsto & 1 \end{array}$$

La composizione di due omomorfismi è ancora un omomorfismo (ed è unico).

$$\begin{array}{rccccc} \alpha\alpha' : \mathbb{N} & \to & \mathbb{N}' & \to & \mathbb{N} \\ 1 & \mapsto & 1' & \mapsto & 1 \end{array} \qquad \begin{array}{rccccc} \alpha'\alpha : \mathbb{N}' & \to & \mathbb{N} & \to & \mathbb{N}' \\ 1' & \mapsto & 1 & \mapsto & 1' \end{array}$$

Risulta:

$$\alpha\alpha' = id_{\mathbb{N}} \quad \alpha'\alpha = id_{\mathbb{N}}$$

Quindi α è un isomorfismo[2] ed $\mathbb{N}$ è l'unico monoide libero generato da un elemento a meno di isomorfismi: $\mathbb{N} \simeq \mathbb{N}'$. □

Proprietà 5.0.22. *Nel monoide libero con un solo generatore l'associatività implica la commutatività.* ◊

[2]Un omomorfismo f prende il nome di isomorfismo se f è una biiezione.

Dimostrazione. Sia $A = \{a\}$, bisogna verificare se A^* è commutativo i.e. se vale $uv = vu$. Si ha il seguente scenario:

$$\underbrace{a \dots a}_{n}\underbrace{a \dots a}_{m} = \underbrace{a \dots a}_{m}\underbrace{a \dots a}_{n}$$

Si dimostra per induzione riconducendosi alla commutatività della somma in $\mathbb{N}$.

$$\begin{array}{lccl} & 0+m & = & m+0 \\ \textbf{Caso base} & = & = & \text{(monoide libero)} \\ & m & m & \end{array}$$

Passo induttivo $h + m = m + h$

Induzione $h + (1 + m) \overset{assoc}{=} (h + 1) + m = s(h) + m = m + s(h) \overset{def.\ s}{=} s(m + h) \overset{comm}{=} s(h + m) \overset{somma}{=} h + s(m)$.

□

Si vuole ora descrivere un monoide libero generato da un insieme finito (o infinito) di generatori.

Proposizione 5.0.23. *Sia A un insieme finito di generatori, il monoide libero generato da A è:*

$$L(A) = A^* = (\{a_1 a_2 \dots a_n \mid a_i \in A,\ \forall i = 1, \dots, n\},\ \cdot\ , \varepsilon)$$

Dimostrazione.

($L(A)$ è generato da A)

Ogni elemento di $L(A)$ è una combinazione finita di elementi di A.

($L(A)$ è un monoide)

Si consideri la *concatenazione* tra parole:

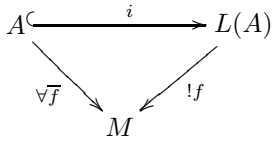
$$\forall a_1 \dots a_n, b_1 \dots b_m \in A \qquad (a_1 \dots a_n) \bullet (b_1 \dots b_n) \overset{def}{=} a_1 \dots a_n b_1 \dots b_m$$

Tale operazione è associativa ed ε è l'elemento neutro.

($L(A)$ è libero)

Vale $A \subseteq L(A)$ poiché ogni elemento di A è una parola di lunghezza unitaria. Si può quindi considerare l'inclusione $i : A \subseteq L(A) \to L(A)$.

Si vuole verificare che per ogni monoide $(M, \bullet, u)$ e per ogni $\overline{f} : A \to M$ il seguente diagramma è commutativo:



Affinché f sia un omomorfismo, bisogna definirla nel modo seguente:

$$\begin{aligned} f(a_1 \dots a_n) &= f(a_1) \bullet \dots \bullet f(a_n) \\ f(\varepsilon) &= u \end{aligned}$$

ed affinché il diagramma sia commutativo deve valere:

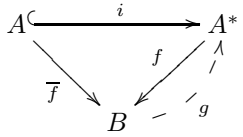
$$f(a_1 \dots a_n) = f(a_1) \bullet \dots \bullet f(a_n) = \overline{f}(a_1) \bullet \dots \bullet \overline{f}(a_n)$$

Quindi, rimane univocamente determinato l'omomorfismo f. □

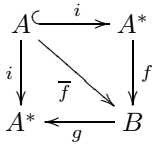
◊

Proposizione 5.0.24 (Unicità A^*)**.** *Il monoide libero A^* è unico a meno di omomorfismi.* ◊

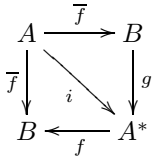
Dimostrazione. Si supponga - per assurdo - che esista un altro monoide libero B.



Allora $\exists! f$ tale che $fi(a) = \overline{f}(a)$. Essendo B libero esiste un'unica g tale che $gf(a) = i(a)$. Si mostra ora che f è l'inversa di g e viceversa.



Quindi esiste un morfismo $A^* \to A^*$ tale che $(gf)i(a) = i(a)$, ed è unico per definizione di monoide libero, che rende commutativo il diagramma. Si ricorda che esiste l'identità, ma allora deve essere unico e quindi $id\, A^* = gf$.



Simmetricamente, $fg = idB$, allora A^* e B sono isomorfi; lo stesso vale prendendo un alfabeto in corrispondenza biunivoca (si è interessati alla cardinalità dell'insieme). Anche l'algebra iniziale è unica. □

Esempio 5.0.25 (Lunghezza di una parola)**.** Sia A^* il monoide libero generato da A. Si può considerare una funzione:

$$\begin{array}{rcl} \overline{l} : A & \to & \mathbb{N} \\ a_i & \mapsto & 1 \end{array}$$

Questa individua un unico omomorfismo l:

$$\begin{array}{rcl} l : L(A) & \stackrel{!}{\to} & \mathbb{N} \\ a_1 \dots a_n & \mapsto & l(a_1) + \dots + l(a_n) = n \end{array}$$

l prende il nome di *lunghezza*. ⊠

Osservazione 5.0.26. Se si considera un monoide non libero, in generale la lunghezza non è ben definita.

Infatti, poiché un monoide non libero è un monoide in cui valgono delle equazioni, se si considerasse e.g. $a_1 a_2 = a_3$, non si sarebbe in grado di definire univocamente la lunghezza (perché non si sa se la parola $a_1 a_2$ ha lunghezza 2 oppure 1). ◊

Osservazione 5.0.27 (Commutatività). In generale, i monoidi liberi non sono commutativi. L'unico monoide libero commutativo è $\mathbb{N}$ in quanto è generato da un solo elemento (si veda la proposizione 5.0.21). ◊

Esempio 5.0.28. Si consideri il monoide A^* generato da $A = \{a_1, a_2\}$. La commutatività dovrebbe essere espressa da un'equazione e.g. $a_1 a_2 = a_2 a_1$, quindi il monoide non sarebbe più libero. ⊠

In generale, un'equazione induce una relazione di equivalenza, quindi un monoide commutativo è ottenibile come quoziente di un monoide libero.

Proposizione 5.0.29. *Un monoide libero risulta ordinato rispetto all'ordinamento per prefisso.* ◊

Dimostrazione. Esercizio. □

Parte II

Teoria dei dominî

Capitolo 6

Continuità

6.1 CPO

Definizione 6.1.1 (Insieme parzialmente ordinato continuo (CPO))**.** *Un* CPO *è un poset* $(D, \sqsubseteq)$ *in cui tutte le catene non vuote* $d_0 \sqsubseteq d_1 \sqsubseteq d_2 \sqsubseteq d_3 \sqsubseteq \ldots$ *hanno minimo comune maggiorante* $\bigsqcup_{n\geq 0} d_n$. ◊

Nel seguito, quando sarà chiaro dal contesto, denoteremo il CPO $(D, \sqsubseteq)$ con D.

Definizione 6.1.2 (Continuità)**.** *Siano* D *ed* E *insiemi parzialmente ordinati, la funzione* f *è* continua *se e solo se è monotona e conserva il minimo comune maggiorante delle catene i.e. per ogni catena* $d_0 \sqsubseteq d_1 \sqsubseteq d_2 .... \in D$*, in* E *deve valere:*

$$f\left(\bigsqcup_{n\geq 0} d_n\right) = \bigsqcup_{n\geq 0} f(d_n)$$

◊

Osservazione 6.1.3. Il minimo comune maggiorante della catena vuota, se esiste, risulta elemento minimo del CPO, $\perp$ (esercizio). ◊

Definizione 6.1.4 (Esattezza)**.** *Siano i poset* D *ed* E *ambedue con elemento minimo, e sia* $\perp$ *l'elemento minimo di* D*.* $f : D \to E$ *è una funzione* esatta *se e solo se* $f(\perp)$ *è l'elemento minimo di* E*.* ◊

Osservazione 6.1.5. Si osservi che se $f : D \to E$ è monotona e $d_0 \sqsubseteq d_1 \sqsubseteq d_2 \ldots \in D$ è una catena, allora applicando la funzione f si ottiene la catena $f(d_0) \sqsubseteq f(d_1) \sqsubseteq f(d_2) ... \in E$.

Inoltre, se d è un minimo comune maggiorante della prima catena, allora $f(d)$ è un comune maggiorante della seconda e quindi è più grande del minimo comune maggiorante della seconda catena. Quindi, se $f : D \to E$ è una funzione monotona tra CPO:

$$\bigsqcup_{n\geq 0} f(d_n) \sqsubseteq f\left(\bigsqcup_{n\geq 0} d_n\right)$$

Allora, utilizzando la proprietà di antisimmetria di $\sqsubseteq$, per provare che una funzione f tra CPO è monotona è sufficiente verificare che, per ogni catena $d_0 \sqsubseteq d_1 \sqsubseteq d_2 \ldots \in D$ vale in E:

$$f\left(\bigsqcup_{n\geq 0} d_n\right) \sqsubseteq \bigsqcup_{n\geq 0} f(d_n)$$

$\Diamond$

Definizione 6.1.6 (Omomorfismo)**.** *Siano $(D, \leq)$ e $(D', \preceq)$ due CPO. Una funzione $f : D \to D'$ è un* omomorfismo *se è monotona e continua.* $\Diamond$

Lemma 6.1.7. *Ogni funzione continua è monotona.* $\Diamond$

Dimostrazione. Sia la funzione continua $f : D \to D'$ e siano $a \leq b$ in D, bisogna provare che $f(a) \preceq f(b)$ in D'. Si consideri allora la catena non decrescente $a \leq b \leq b \leq b \leq \ldots$, essa ha chiaramente come minimo comune maggiorante b i.e. $\sqcup\{a, b\} = b$.

Quindi, per definizione di continuità:

$$\sqcup\{f(a), f(b)\} = f(\sqcup\{a, b\}) = f(b) \text{ in } D'$$

i.e. $f(a) \preceq f(b)$. □

Lemma 6.1.8 (Intreccio di catene)**.** *Sia $(D, \leq)$ un CPO e siano $x_1 \leq x_2 \leq x_3 \leq \ldots$ e $y_1 \leq y_2 \leq y_3 \leq \ldots$ due catene di D tali che:*

1. $\forall x_i \; \exists y_j \;\; x_i \leq y_j$
2. $\forall y_i \; \exists x_j \;\; y_i \leq x_j$

Allora $\sqcup x_i = \sqcup y_j$. $\Diamond$

Dimostrazione. Bisogna dimostrare che $\sqcup y_j \leq \sqcup x_i$ e $\sqcup x_i \leq \sqcup y_j$, da questo si ha $\sqcup y_j = \sqcup x_i$.

Si consideri un arbitrario y_j, dalle ipotesi segue che esiste un indice k tale che $y_j \leq x_k$; poiché $\sqcup x_i$ è il minimo comune maggiorante degli x_i, allora $y_j \leq$

$x_k \leq \sqcup x_i$. Essendo la precedente disuguaglianza vera per ogni indice j, segue che $\sqcup y_j \leq \sqcup x_i$. Allo stesso modo si può provare che $\sqcup x_i \leq \sqcup y_j$, e quindi $\sqcup x_i = \sqcup y_j$. □

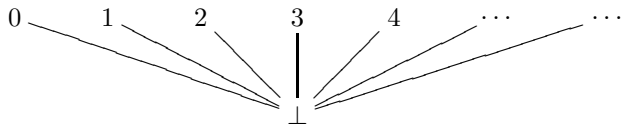
6.2 Dominî

Definizione 6.2.1 (Dominio)**.** *Un* dominio $(D, \sqsubseteq)$ *è un CPO che possiede elemento minimo* $\perp$*:*

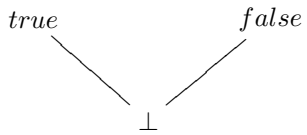
$$\forall d \in D \;\; \perp \sqsubseteq d$$

◊

Esempio 6.2.2 (Dominî)**.** Un esempio possibile è il dominio piatto dei numeri naturali, $\mathbb{N}_\perp$:



oppure il dominio piatto dei valori booleani, $\mathbb{B}_\perp$:



⊠

Esempio 6.2.3 (Dominî di funzioni parziali)**.** L'insieme di tutte le funzioni parziali $X \rightharpoonup Y$ può essere trasformato in un dominio nel modo seguente.

Insieme sottostante Si considerano tutte le funzioni parziali f, con dominio di definizione $dom(f) \subseteq X$ che inviano valori in Y.

Ordine parziale $f \sqsubseteq g \;\Leftrightarrow\; dom\,(f) \subseteq dom\,(g) \quad \text{e} \quad \forall x \in dom\,(f) \quad f\,(x) = g\,(x)$

Minimo comune maggiorante di una catena Il minimo comune maggiorante della catena $f_0 \sqsubseteq f_1 \sqsubseteq f_2 \sqsubseteq f_3 \sqsubseteq ...$ è la funzione parziale f tale che:

$$dom\,(f) = \bigcup_{n \geq 0} dom\,(f_n)$$

$$f\,(x) = \begin{cases} f_n\,(x) & \text{se } x \in dom\,(f_n)\,, \text{ per un qualche } n \\ indefinita & \text{altrimenti} \end{cases}$$

L'elemento minimo $\perp$ è la funzione parziale totalmente indefinita.

$\boxtimes$

Si introducono ora due funzioni continue su dominî piatti, che saranno necessarie per definire la semantica denotazionale di PCF.

Proposizione 6.2.4 ($f_\perp$)**.** *Sia $f : X \rightharpoonup Y$ una funzione parziale tra due insiemi, allora $f_\perp : X_\perp \to Y_\perp$*

$$f_\perp(d) = \begin{cases} f(d) & \text{se } d \in X \text{ e } f \text{ è definita in } d \\ \perp & \text{se } d \in X \text{ e } f \text{ non è definita in } d \\ \perp & \text{se } d = \perp \end{cases}$$

definisce una funzione continua tra i corrispondenti domini piatti. $\lozenge$

Proposizione 6.2.5. *Per ogni dominio D la funzione:*

$$if : \mathbb{B}_\perp \times (D \times D) \to D$$

$$if(x,(d,d')) \stackrel{def}{=} \begin{cases} d & \text{se } x = true \\ d' & \text{se } x = false \\ \perp_D & \text{se } x = \perp \end{cases}$$

è continua. $\lozenge$

6.3 Operazioni tra dominî

Definizione 6.3.1 (Prodotto binario tra CPO)**.** *Il* prodotto *di due CPO $(D_1, \sqsubseteq_1)$ e $(D_2, \sqsubseteq_2)$ è un CPO $(D_1 \times D_2, \sqsubseteq)$ che ha come insieme sottostante*

$$D_1 \times D_2 = \{(d_1, d_2) \mid d_1 \in D_1,\ d_2 \in D_2\}$$

e ordine parziale $\sqsubseteq$ definito come:

$$(d_1, d_2) \sqsubseteq (d_1', d_2') \stackrel{def}{\Leftrightarrow} d_1 \sqsubseteq_1 d_1' \text{ e } d_2 \sqsubseteq_2 d_2'$$

I minimi comuni maggioranti delle catene sono calcolati componente per componente

$$\bigsqcup_{n \geq 0} (d_{1,n}, d_{2,n}) = \left(\bigsqcup_{i \geq 0} d_{1,i}, \bigsqcup_{j \geq 0} d_{2,j} \right)$$

$\lozenge$

Proposizione 6.3.2 (Proiezioni e accoppiamento)**.** *Siano i CPO D_1 e D_2. Le proiezioni*

$$\begin{array}{rcl} \pi_1 \ : \ D_1 \times D_2 & \to & D_1 \\ \pi_1(d_1, d_2) & \mapsto & d_1 \end{array} \qquad \begin{array}{rcl} \pi_2 \ : \ D_1 \times D_2 & \to & D_2 \\ \pi_2(d_1, d_2) & \mapsto & d_2 \end{array}$$

sono funzioni continue. Se $f_1 : D \to D_1$ e $f_2 : D \to D_2$ sono funzioni continue da un CPO D, allora

$$\begin{array}{rcl} \langle f_1, f_2 \rangle \ : \ D & \to & D_1 \times D_2 \\ \langle f_1, f_2 \rangle(d) & \mapsto & (f_1(d), f_2(d)) \end{array}$$

è continua. ◊

Dimostrazione. La continuità di queste funzioni segue immediatamente dalla caratterizzazione dei minimi comuni maggioranti delle catene in $D_1 \times D_2$ fornita nella definizione (6.3.1). □

Proposizione 6.3.3 (Funzioni monotone e continue tra CPO)**.** *Siano i CPO D, E ed F, una funzione $f : D \times E \to F$ è* monotona *se e solo se è monotona in ogni argomento separatamente i.e.*

$$\begin{array}{ll} \forall d, d' \in D, \ e \in E & d \sqsubseteq d' \ \Rightarrow \ f(d, e) \sqsubseteq f(d', e) \\ \forall d \in D, e, e' \in E & e \sqsubseteq e' \ \Rightarrow \ f(d, e) \sqsubseteq f(d, e') \end{array}$$

Inoltre la funzione f è continua *se e solo se preserva i minimi comuni maggioranti delle catene in ogni argomento separatamente i.e.*

$$f\left(\bigsqcup_{m \geq 0} d_m, e\right) = \bigsqcup_{m \geq 0} f(d_m, e)$$

$$f\left(d, \bigsqcup_{n \geq 0} e_n\right) = \bigsqcup_{n \geq 0} f(d, e_n)$$

Dimostrazione.

($\Leftarrow$)
Se $d \sqsubseteq d'$ allora:

$$(d, e) \sqsubseteq (d', e)$$

e

$$\left(\bigsqcup_{m \geq 0} d_m, e\right) = \bigsqcup_{m \geq 0} (d_m, e)$$

e analogamente per per l'argomento a destra.

($\Rightarrow$)

Supponiamo che la funzione f sia monotona in ogni argomento separatamente. Allora dati $(d, e), (d', e') \in D \times E$:

$$(d, e) \sqsubseteq (d', e')$$

Dalla definizione di ordine parziale sul prodotto binario si ha che per $d, d' \in D$ ed $e, e' \in E$:

$$d \sqsubseteq d' \text{ in } D$$
$$e \sqsubseteq e' \text{ in } E$$

$$f\,(d, e) \sqsubseteq f\,(d', e) \qquad \text{(monotonia primo argomento)}$$

$$\sqsubseteq f\,(d', e') \qquad \text{(monotonia secondo argomento)}$$

$$f\,(d, e) \sqsubseteq f\,(d', e') \qquad \text{(transitività)}$$

come richiesto per la monotonia di f.

Ora si supponga che f sia continua in ogni argomento separatamente. Allora, data una catena, nel prodotto binario si ha:

$$(d_0, e_0) \sqsubseteq (d_1, e_1) \sqsubseteq (d_2, e_2) \sqsubseteq (d_3, e_3) \sqsubseteq \ldots$$

(proposizione 6.3.1)

$$f\left(\bigsqcup_{n\geq 0} (d_n, e_n)\right) = f\left(\bigsqcup_{i\geq 0} d_i, \bigsqcup_{j\geq 0} e_j\right)$$

$$= \bigsqcup_{i\geq 0} f\left(d_i, \bigsqcup_{j\geq 0} e_j\right) \qquad \text{(continuità primo argomento)}$$

$$= \bigsqcup_{i\geq 0} \left(\bigsqcup_{j\geq 0} f\,(d_i, e_j)\right) \qquad \text{(continuità secondo argomento)}$$

$$= \bigsqcup_{n\geq 0} f\,(d_n, e_n) \qquad \text{(lemma (6.3.4))}$$

Come richiesto per la continuità di f. □

◊

Lemma 6.3.4 (Diagonalizzazione delle catene doppie)**.** *Sia D un CPO, se la famiglia di elementi $d_{m,n} \in D$ $(m, n \geq 0)$ soddisfa:*

$$m \leq m' \wedge n \leq n' \;\Rightarrow\; d_{m,n} \sqsubseteq d_{m',n'} \tag{6.1}$$

allora

$$\bigsqcup_{n\geq 0} d_{0,n} \sqsubseteq \bigsqcup_{n\geq 0} d_{1,n} \sqsubseteq \bigsqcup_{n\geq 0} d_{2,n} \sqsubseteq \bigsqcup_{n\geq 0} d_{3,n} \sqsubseteq ...$$

e

$$\bigsqcup_{m\geq 0}\left(\bigsqcup_{n\geq 0} d_{m,n}\right) = \bigsqcup_{k\geq 0} d_{k,k} = \bigsqcup_{n\geq 0}\left(\bigsqcup_{m\geq 0} d_{m,n}\right)$$

Graficamente:

$$\begin{array}{cccccccc}
\sqcup d_{m,0} & \sqcup d_{m,1} & \dots & \sqcup d_{k,k} & \dots & \sqcup d_{1,n} & \sqcup d_{0,n} \\
 & \dots & & & & \dots & \\
\dots & & \dots & \dots & \dots & & \dots \\
 & \dots & & & \dots & & \\
\dots & \sqcup d_{2,0} & & \sqcup d_{1,1} & & \sqcup d_{0,2} & \dots \\
 & & \sqcup d_{1,0} & & \sqcup d_{0,1} & & \\
 & & & \sqcup d_{0,0} & & &
\end{array}$$

La colonna al centro rappresenta $\bigsqcup_{k\geq 0} d_{k,k}$, la diagonale. ◊

Dimostrazione. Per la dimostrazione si utilizzano le proprietà viste in precedenza sul minimo comune maggiorante di una catena.

Se $m \leq m'$ allora:

$$d_{m,n} \sqsubseteq d_{m',n} \qquad \text{(proprietà (6.1))}$$

$$\sqsubseteq \bigsqcup_{n'\geq 0} d_{m',n'} \qquad \text{(proprietà (2.1))}$$

$$\forall n \geq 0 \quad \bigsqcup_{n\geq 0} d_{m,n} \sqsubseteq \bigsqcup_{n'\geq 0} d_{m',n'} \qquad \text{(proprietà (2.2))}$$

Così si ha una catena di minimi comuni maggioranti:

$$\bigsqcup_{n\geq 0} d_{0,n} \sqsubseteq \bigsqcup_{n\geq 0} d_{1,n} \sqsubseteq \bigsqcup_{n\geq 0} d_{2,n} \sqsubseteq \bigsqcup_{n\geq 0} d_{3,n} \sqsubseteq ...$$

Si può costruire il suo minimo comune maggiorante $\bigsqcup_{m\geq 0}\bigsqcup_{n\geq 0} d_{m,n}$

Utilizzando due volte la proprietà (2.1) si ha:

$$d_{k,k} \sqsubseteq \bigsqcup_{n\geq 0} d_{k,n} \sqsubseteq \bigsqcup_{m\geq 0} \bigsqcup_{n\geq 0} d_{m,n}$$

per ogni $k \geq 0$, e quindi per la proprietà (2.2):

$$\bigsqcup_{k\geq 0} d_{k,k} \sqsubseteq \bigsqcup_{m\geq 0} \bigsqcup_{n\geq 0} d_{m,n} \tag{6.2}$$

Al contrario, per ogni $m, n \geq 0$, si ha che:

$$\begin{aligned} d_{m,n} &\sqsubseteq d_{\max\{m,n\},\max\{m,n\}} && \text{(proprietà (6.1))} \\ &\sqsubseteq \bigsqcup_{k\geq 0} d_{k,k} && \text{(proprieta (2.1))} \end{aligned}$$

Ora applicando due volte la proprietà (2.2) si ha:

$$\bigsqcup_{m\geq 0} \bigsqcup_{n\geq 0} d_{m,n} \sqsubseteq \bigsqcup_{k\geq 0} d_{k,k} \tag{6.3}$$

Combinando la (6.2) e la (6.3) con l'antisimmetria di $\sqsubseteq$ si ha l'uguaglianza desiderata.

Si ottiene l'uguaglianza addizionale usando gli stessi argomenti, ma scambiando i ruoli di m ed n. □

Definizione 6.3.5 (Prodotti indiciati)**.** *Sia I un insieme, si supponga che ad ogni $i \in I$ corrisponda un CPO $(D_i, \sqsubseteq_i)$. Il* prodotto *dell'intera famiglia di CPO ha*

- *insieme sottostante uguale al prodotto cartesiano $\prod_{i\in I} D_i$ di insiemi D_i ed è costituito da tutte le funzioni p definite su I e tale che il valore di p per ogni $i \in I$ è un elemento $p(i) \in D_i$ del CPO D_i*
- *ordine parziale $\sqsubseteq$ definito da*

  $$p \sqsubseteq p' \overset{def}{\Leftrightarrow} \forall i \in I \;\; p(i) \sqsubseteq_i p'(i)$$

  Come per il prodotto binario (caso particolare in cui I è un insieme di due elementi) i minimi comuni maggioranti in $(\prod_{i\in I} D_i, \sqsubseteq)$ possono essere calcolati componente per componente: se $p_0 \sqsubseteq p_1 \sqsubseteq p_2 \sqsubseteq \ldots$ è una catena nel CPO prodotto, il suo minimo comune maggiorante è la funzione che invia ogni $i \in I$ nel minimo comune maggiorante in D_i della catena $p_0(i) \sqsubseteq p_1(i) \sqsubseteq p_2(i) \sqsubseteq \ldots$, perciò

  $$(\bigsqcup_{n\geq 0} p_n)(i) = \bigsqcup_{n\geq 0} p_n(i) \qquad (i \in I)$$

In particolare, per ogni $i \in I$ l'i-esima funzione di proiezione

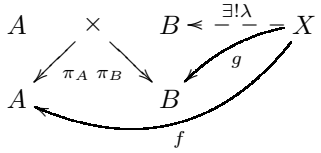
$$\begin{array}{rccc} \pi_i & : & \prod_{i' \in I} D_{i'} & \to & D_i \\ & & \pi_i(p) & \mapsto & p(i) \end{array}$$

è continua. Se tutti i D_i sono dominî, allora anche il loro prodotto è un dominio, avente come elemento minimo la funzione che invia ogni $i \in I$ nell'elemento minimo di D_i.

$\Diamond$

Questa nozione di prodotto può essere considerata un'istanza di una definizione generale che ora riportiamo.

Definizione 6.3.6 (Prodotto di due strutture)**.** *Siano A e B due strutture, il* prodotto *$A \times B$ è determinato dalle sue proiezioni, che sono funzioni su A e B che ne conservano la struttura. In più vale che ogni altra struttura con due funzioni come nel caso precedente, si fattorizza in modo unico attraverso il prodotto.*

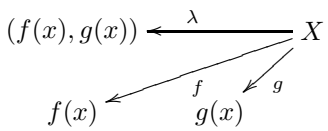


Esiste, cioè, un unico λ, funzione che conserva la struttura, tale che il diagramma è commutativo, ossia $f = \pi_A\lambda$ e $g = \pi_B\lambda$. $\Diamond$

Proposizione 6.3.7. *Il prodotto di due strutture, se esiste, è unico a meno di isomorfismi.* $\Diamond$

Dimostrazione. Si supponga che X sia un altro prodotto, allora deve esistere μ da $A \times B$ in X, con $\pi_A = f\mu$, $\pi_B = g\mu$. Sostituendo: $\pi_A = f\mu = \pi_A\lambda\mu$, $\pi_B = g\mu = \pi_B\lambda\mu$, ma π_A si fattorizza in modo unico rispetto a se stesso. Quindi $\lambda\mu = id_{A\times B}$, $\mu\lambda = id_X$, allora $A \times B$ e X sono isomorfi e il prodotto è univocamente determinato. □

Osservazione 6.3.8 (Importanza dell'esistenza delle proiezioni). λ è anche conosciuta come $\langle f, g\rangle$, si ha $\pi_A\langle f, g\rangle = f$ e $\pi_B\langle f, g\rangle = g$. Sia $x \in X$, f invia x in $f(x)$ e g invia x in $g(x)$. Per essere commutativo, il prodotto deve essere $(f(x), g(x))$ e λ è la funzione coppia.



◊

Proposizione 6.3.9 (Prodotto tra poset)**.** *Siano i poset* $(A, \leq_A)$ *e* $(B, \leq_B)$*, si definisce il poset* $(A \times B, \leq)$ *indicando un ordinamento con proiezioni monotone. Quindi, sia* $(A \times B, \leq)$ *dove*

$$(a, b) \leq (a', b') \text{ se e solo se } (a \leq_A a') \wedge (b \leq_B b')$$

◊

Dimostrazione. Sappiamo che le proiezioni sono monotone. Facendo riferimento al grafo in (6.3.6) bisogna verificare che, considerando X, esiste ed è unica la funzione coppia $\langle f, g \rangle$ che rende commutativo il diagramma. Se $x \leq y$ allora $((f(x), g(x)) \leq (f(y), g(y)))$ poiché sia f che g sono monotone e quindi $f(x) \leq f(y)$ e $g(x) \leq g(y)$ per definizione di ordinamento prodotto. Quindi anche il prodotto tra poset è un poset, a meno di isomorfismi. □

Osservazione 6.3.10 (Ordinamento prodotto). Se A e B fossero ordini totali allora $A \times B$ non lo sarebbe poiché questo è un ordinamento prodotto. Invece l'ordinamento lessicografico sarebbe un ordine totale ma non soddisferebbe le condizioni del prodotto prodotto. ◊

Abbiamo già considerato il prodotto di due CPO che risulta un poset con minimo e minimo comune maggiorante per ogni catena e tale che le proiezioni siano continue. Analogamente si otterrà il prodotto di dominî, richiedendo in più l'esistenza di un minimo e la sua conservazione, cioè l'esattezza delle proiezioni.

Proposizione 6.3.11 (Prodotto di due dominî)**.** *Siano i CPO* $(A, \leq_A)$ *e* $(B, \leq_B$ *) con i minimi* $\perp_A$ *e* $\perp_B$*, per definizione di ordinamento* $(\perp_A, \perp_B)$ *è il minimo di* $A \times B$ *i.e.* $\forall (a, b) \in A \times B \ (\perp_A, \perp_B) \leq (a, b)$*.* ◊

Dimostrazione. Si consideri una catena in A e una catena in B: dobbiamo mostrare che la coppia dei minimi comuni maggioranti è il minimo comune maggiorante delle coppie; il minimo comune maggiorante esiste in quanto

$$a_0 \leq_A a_1 \leq_A a_2 \leq_A \ldots \leq_A \sqcup_i a_i$$

$$b_0 \leq_B b_1 \leq_B b_2 \leq_B \ldots \leq_B \sqcup_i b_i$$

Date le catene precedenti, la seguente è ancora una catena per definizione di ordinamento:

$$(a_0, b_0) \leq (a_1, b_1) \leq (a_2, b_2) \leq \ldots \leq (\sqcup_i a_i, \sqcup_i b_i)$$

Vale anche il viceversa, per definizione di ordinamento si possono estrapolare due catene: una catena in $A \times B$ corrisponde a una catena in A e una in B (per la monotonia delle proiezioni).

La coppia $(\sqcup_i a_i, \sqcup_i b_i)$ è il minimo comune maggiorante della catena: è un comune maggiorante $(a_k, b_k) \leq (\sqcup_i a_i, \sqcup_i b_i)$ poiché $a_k \leq \sqcup_i a_i$ e $b_k \leq \sqcup_i b_i$.

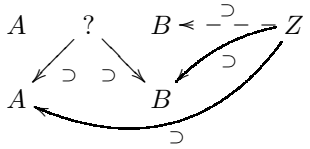
Inoltre è il minimo: sia $c \in A \times B$ tale che maggiori la catena i.e. $\forall k\ (a_k, b_k) \leq c$. Poiché $c \in A \times B$ si possono considerare le proiezioni $\pi_A(c) = c_1$ e $\pi_B(c) = c_2$. Bisogna verificare che $\forall k\ a_k \leq c_1$ e $\forall k\ b_k \leq c_2$.

Poiché c_1 maggiora a_i e c_2 maggiora b_i si ha $c_1 \geq \sqcup_i a_i$ e $c_2 \geq \sqcup_i b_i$, quindi $(\sqcup_i a_i, \sqcup_i b_i) \leq (c_1, c_2) = c$.

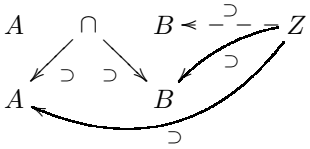
$(\sqcup_i a_i, \sqcup_i b_i)$ è il minimo perché c maggiora la catena, quindi il minimo comune maggiorante è $\sqcup_i (a_i, b_i)$.

Si può concludere che il prodotto – così definito – tra due dominî è un dominio. □

Osservazione 6.3.12 (Intersezione). Consideriamo ora un'altra situazione di prodotto tra insiemi. Si definisce $A \times B$ non attraverso generiche funzioni, ma mediante inclusioni. Sia $(\mathcal{P}(X), \subseteq)$ un poset e siano A e B due sottoinsiemi di X. Ci si chiede se esiste un'inclusione che rende commutativo il seguente diagramma (dove '?' indica l'inclusione da trovare).



L'intersezione soddisfa le condizioni richieste:



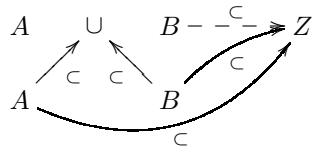
definita a meno di identità. In generale per ogni reticolo (non solo $\mathcal{P}(X)$ e algebre di Boole) il prodotto è

$$a \cap b \leq a, b \quad \forall z \leq a, b \quad z \leq a \cap b$$

◊

Partendo da quest'ultimo caso, consideriamo la situazione duale, che, in un ordinamento si riduce ad invertire l'ordine.

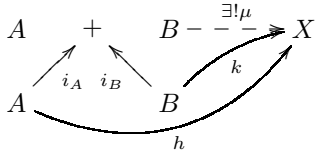
Proposizione 6.3.13 (Unione)**.** *La* somma *(unione) soddisfa la condizione:*



$$a \cup b \geq a, b \quad \forall z \geq a, b \quad z \geq a \cup b$$

l'unione è il minimo comune maggiorante. Si ottiene analogamente la nozione generale di coprodotto *(i.e. il duale).*

Definizione 6.3.14 (Somma)**.** *Dati due oggetti A e B, la loro somma o coprodotto, $A + B$ è codominio di due morfismi di struttura i_A e i_B, rispettivamente da A e da B, in modo che in ogni altro oggetto X con le stesse proprietà, esista unico μ che renda commutativo il seguente diagramma; cioè tale che $h = \mu i_A$ e $k = \mu i_B$. i_A, i_B sono dette* iniezioni canoniche*.*



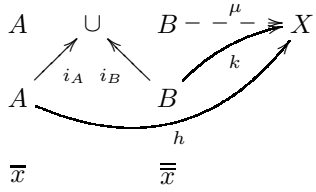
◊

Proposizione 6.3.15. *La somma è unica a meno di isomorfismi.* ◊

Dimostrazione. La dimostrazione è duale di quella del prodotto. □

Analizziamo ora chi sia $A + B$ negli insiemi.

Si prova con l'unione:

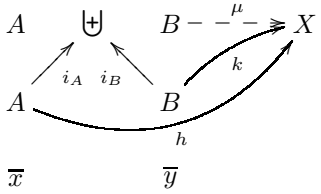


Per le iniezioni si considera l'inclusione. Con h, k bisogna costruire μ: si prende $x \in A \cup B$ tale che se $\overline{x} \in A$ allora $x \in A \cup B = i_A(\overline{x})$ e $\mu i_A(\overline{x} = h(\overline{x}))$; analogamente per B.

Il problema è che non si sa dove inviare gli elementi in $A \cap B$. Tuttavia, la definizione è valida se si considera l'unione disgiunta $A \biguplus B$: se $x \in A$ è inviato in $h(\overline{x})$ e se $x \in B$ è inviato in $k(\overline{\overline{x}})$. La funzione è unica perché altrimenti verrebbe meno la commutatività del grafo.

Siccome si è interessati agli insiemi con struttura, analizziamo cosa succede con i poset.

Esempio 6.3.16 (Unione poset)**.** Si definisce un ordinamento su $A \biguplus B$ in modo che le iniezioni ora risultino monotone.



Per monotonia deve valere, in $A \biguplus B$, $x \leq_A x'$,se x e x' in $A \biguplus B$ provengono da A o da B, i.e. $i_A(\overline{x}), i_A(\overline{x'})$ deve valere

$$x \leq_A x' \Leftrightarrow \overline{x} \leq_A \overline{x'}$$

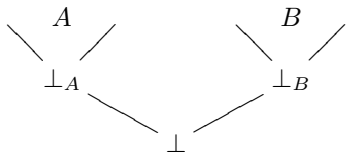
Rimane da scegliere tra $i_A(\overline{x}) \overset{?}{\leq} i_B(\overline{y})$ e $i_A(\overline{x}) \overset{?}{\geq} i_B(\overline{y})$, ma essendo un ordinamento parziale possiamo non scegliere nulla, mantenendo l'ordinamento da A e da B e non aggiungendo altro.

Bisogna verificare che $\leq$ sia un ordine parziale, la proprietà riflessiva vale da A e B; l'antisimmetria è verificata poiché A e B sono separati e le catene non saranno mai trasversali (o sono in A o sono in B); così per la transitività. Le iniezioni sono monotone per costruzione; che esista e sia unica μ si deriva dalla costruzione, in più è monotona perché è obbligata solo a conservare l'ordinamento o di A o di B. In pratica μ è h per gli elementi di A e k per gli elementi di B.

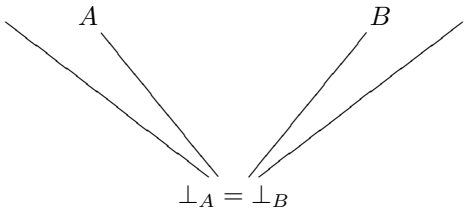
La scelta di non aggiungere niente è l'unica possibile, aggiungendo qualcosa non si sarebbe potuto verificare che μ conservasse la transitività. Questo è l'*ordinamento somma*: ordina separatamente il primo ed il secondo insieme.

Esempio 6.3.17 (Somma di dominî)**.** Siano A e B due dominî, bisogna definire la loro somma in modo che le funzioni i_A, i_B siano esatte. Essendo dominî, esistono $\bot_A$ e $\bot_B$ elementi minimi, ma qual è il minimo in $A \biguplus B$? Si hanno due possibilità:

1. aggiungere $\bot$ come minimo



2. far coincidere $\perp_A, \perp_B$ i.e. dire che $\perp_A = \perp_B$ è il minimo



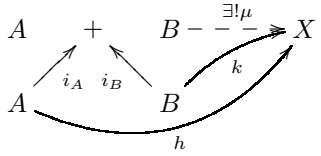
Nel primo caso i_A è id_A, tranne che per $\perp_A$ che è inviato in $\perp$. Anche per le catene tale approccio è corretto: le funzioni i_A e i_B sono continue. Nel secondo caso $\perp_A$ è inviato in $\perp_A = \perp_B$.

Analizziamo il comportamento rispetto alle funzioni. Per ogni h, k esiste μ, nel primo caso si ha che $\perp_A$ e $\perp_B$ sono inviati in $\perp_X$. Analizziamo μ:

$$\begin{aligned} \mu(a) &= h(a) \\ \mu(b) &= k(b) \\ \mu(\perp) &= \mu(\perp_X) \\ \mu(\perp_A) &= \perp = \mu(\perp_B) \end{aligned}$$

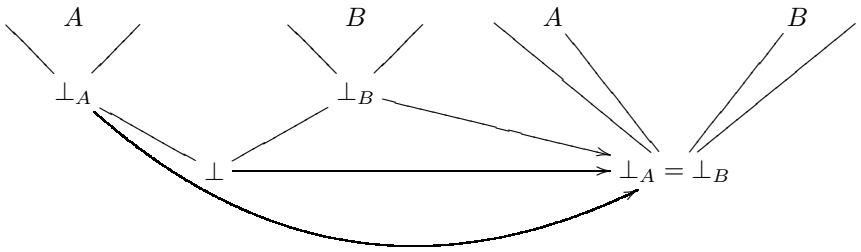
Nel secondo caso la definizione è la stessa.

A questo punto bisogna scegliere tra aggiungere $\perp$ o imporre $\perp_A = \perp_B$

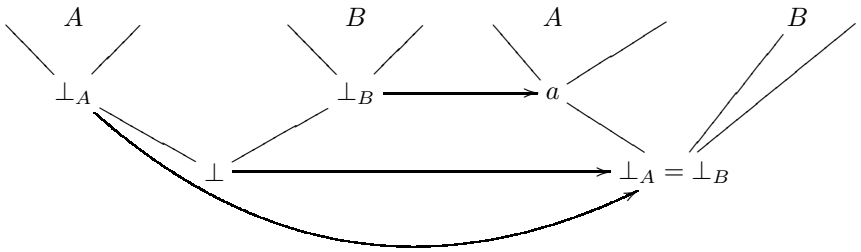


Si ha che $i_A \sqcup_i x_i = \sqcup_i i_A x_i$ è vera per entrambe le soluzioni.

Il problema è quindi nella scelta di μ, perché cerchiamo la soluzione universale. Si consideri la 1) come soluzione $A + B$ e la 2) come X e vediamo perché non è una soluzione valida. Ciò che è scorretto è l'unicità: i due vecchi minimi $\perp_A$ e $\perp_B$ non vengono considerati, c'è più di un morfismo che rende commutativo il diagramma:



e.g.



Si osservi che la costruzione è analoga a quella per il DFA che riconosce l'unione di due linguaggi regolari, anche se sarebbe meglio unificare gli stati iniziali. Quindi la soluzione valida è la 2) [1]. ⊠

6.4 Dominî di funzioni

L'insieme delle funzioni continue tra CPO/dominî può essere trasformato in un CPO.

Proposizione 6.4.1. *Siano $(D, \sqsubseteq_D)$ e $(E, \sqsubseteq_E)$ due CPO, il CPO funzione $(D \to E, \sqsubseteq)$ ha insieme sottostante*

$$D \to E \stackrel{def}{=} \{f \mid f : D \to E \text{ è una funzione continua}\}$$

ed ordine parziale

$$f \sqsubseteq f' \stackrel{def}{\Leftrightarrow} \forall d \in D \;\; f(d) \sqsubseteq_E f'(d)$$

I minimi comuni maggioranti delle catene sono calcolati "argomento per argomento" (usando i minimi comuni maggioranti in E):

$$\left(\bigsqcup_{n \geq 0} f_n\right)(d) = \bigsqcup_{n \geq 0} f_n(d)$$

[1] Da dimostrare per esercizio.

Se E è un dominio allora anche $D \to E$ è un dominio e $\bot_{D\to E}(d) = \bot_E$ per tutti gli elementi $d \in D$. $\Diamond$

Dimostrazione. Bisogna dimostrare che il minimo comune maggiorante della catena di funzioni $\bigsqcup_{n\geq 0} f_n$, è continuo. Si consideri una catena nell'insieme D.

$$\left(\bigsqcup_{n\geq 0} f_n\right)\left(\left(\bigsqcup_{m\geq 0} d_m\right)\right) = \bigsqcup_{n\geq 0}\left(f_n\left(\bigsqcup_{m\geq 0} d_m\right)\right) \quad \text{(definizione di } \bigsqcup_{n\geq 0} f_n\text{)}$$

$$f_n = \bigsqcup_{n\geq 0}\left(\bigsqcup_{m\geq 0} f_n(d_m)\right) \quad \text{(continuità di ogni } f_n\text{)}$$

$$= \bigsqcup_{n\geq 0}\left(\bigsqcup_{m\geq 0} f_n(d_m)\right) \quad \text{(proprietà (6.2))}$$

$$\bigsqcup_{n\geq 0} f_n = \bigsqcup_{m\geq 0}\left(\left(\bigsqcup_{n\geq 0} f_n\right)(d_m)\right) \quad \text{(definizione di } \bigsqcup_{n\geq 0} f_n\text{)}$$

□

6.5 Evaluation e Currying

Proposizione 6.5.1 (Continuità *ev*)**.** *Siano D ed E due CPO, la funzione:*

$$\begin{aligned} ev : (D \to E) \times D &\to E \\ ev(f,d) &\mapsto f(d) \end{aligned}$$

è continua. $\Diamond$

Dimostrazione.

$$ev\left(\bigsqcup_{n\geq 0}(f_n, d_n)\right) = ev\left(\bigsqcup_{i\geq 0} f_i, \bigsqcup_{j\geq 0} d_j\right) \qquad \text{(minimi comuni maggioranti}$$

nei prodotti sono calcolati componente per componente)

$$= \left(\bigsqcup_{i\geq 0} f_i\right)\left(\bigsqcup_{j\geq 0} d_j\right) \qquad \text{(definizione di } ev\text{)}$$

$$= \bigsqcup_{i\geq 0} f_i \left(\bigsqcup_{j\geq 0} d_j\right) \qquad \text{(minimi comuni maggioranti nei CPO}$$

funzione sono calcolati argomento per argomento)

$$f_i = \bigsqcup_{i\geq 0}\bigsqcup_{j\geq 0} f_i(d_j) \qquad \text{(continuità di ogni } f_i\text{)}$$

$$= \bigsqcup_{n\geq 0} f_n(d_n) \qquad \text{(lemma (6.3.4))}$$

$$= \bigsqcup_{n\geq 0} ev(f_n, d_n) \qquad \text{(definizione di } ev\text{)}$$

Quindi si è dimostrato che la funzione ev è continua. □

Proposizione 6.5.2 (Continuità cur)**.** *Sia $f : D' \times D \to E$ una funzione continua, per ogni $d' \in D'$ con D' un CPO. La funzione $d \in D \mapsto f(d', d)$ è continua e determina un elemento del CPO funzione $D \to E$ chiamato $cur(f)(d')$[2].*

Allora

$$\begin{array}{rcl} cur(f) : D' & \to & (D \to E) \\ cur(f)(d') & \mapsto & \lambda d \in D \quad f(d', d) \end{array}$$

è una funzione continua. ◊

Dimostrazione. La continuità di ogni $cur(f)(d')$ e quindi di $cur(f)$ segue dal fatto che ogni minimo comune maggiorante delle catene $D_1 \times D_2$ può essere calcolato componente per componente. □

Quindi si è dimostrato che esiste una funzione ev che è continua e definita in $(D \to E) \times D \to E$. Inoltre si è dimostrato che, data un'altra funzione $f : D' \times D \to E$ esiste ed è unica la funzione continua $cur(f)$ tale che:

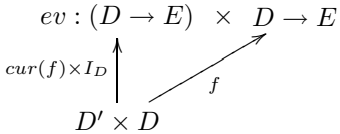


Tabella 6.1: Haskell Brooks Curry

Tratto da [Wik06]. *Haskell Brooks Curry* (12 settembre 1900 - 1 settembre 1982) fu un matematico e logico americano. Nato a Millis, Massachusetts, figlio dell'educatore Samuel Silas Curry, studiò all'*università di Harvard* e ricevette il dottorato a Göttingen nel 1930, sotto la supervisione di *David Hilbert*. Insegnò a *Harvard*, *Princeton*, e poi dal 1929 per 35 anni all'*università statale della Pennsylvania*. Nel 1942 espose il paradosso di Curry. Nel 1966 divenne professore di matematica all'*università di Amsterdam*.
Il lavoro principale di Curry fu in logica matematica, specialmente in logica combinatoria, le fondamenta per un linguaggio di programmazione funzionale.
I linguaggi funzionali *Haskell* e *Curry* sono stati così chiamati in suo onore, così come il concetto di curryficazione in logica combinatoria, lambda calcolo e programmazione funzionale.

6.6 Punto fisso e minimo punto prefisso

Definizione 6.6.1 (Punto fisso)**.** *Un* punto fisso *per una funzione* $f : D \to D$ *è un elemento* $d \in D$ *tale che* $f(d) = d$. ◊

Se D è un insieme parzialmente ordinato, si può considerare la nozione più debole di *punto prefisso*.

Definizione 6.6.2 (Punto prefisso)**.** *Sia* $(D, \sqsubseteq)$ *un insieme parzialmente ordinato e sia* $f : D \to D$ *una funzione, un elemento* $d \in D$ *è un* punto prefisso *di* f *se* $f(d) \sqsubseteq d$. ◊

Definizione 6.6.3 (Minimo punto prefisso)**.** *Il* minimo punto prefisso *di* f *(denotato con* $fix(f)$*), se esiste, è specificato (univocamente) dalle seguenti proprietà:*

$$f(fix(f)) \sqsubseteq fix(f) \tag{6.4}$$

$$\forall d \in D \quad f(d) \sqsubseteq d \;\Rightarrow\; fix(f) \sqsubseteq d \tag{6.5}$$

◊

[2] i.e. la versione "currificata" di f.

Proposizione 6.6.4. *Sia D un insieme parzialmente ordinato e $f : D \to D$ una funzione che possiede un minimo punto prefisso $fix\,(f)$. Essendo la funzione f monotona, $fix\,(f)$ è un particolare punto fisso di f (ed è il più piccolo elemento dell'insieme dei punti fissi di f).* ◊

Dimostrazione. Per definizione $fix\,(f)$ soddisfa la proprietà (6.4). Applicando f ad entrambe le espressioni di (6.4) ed essendo f monotona:

$$f\,(f\,(fix\,(f))) \sqsubseteq f\,(fix\,(f))$$

Applicando la proprietà (6.5) si ha:

$$fix\,(f) \sqsubseteq f\,(fix\,(f))$$

Combinando il risultato con la proprietà (6.4) e l'antisimmetria dell'ordinamento parziale $\sqsubseteq$ si ottiene $f\,(fix\,(f)) = fix\,(f)$, come richiesto. □

Proposizione 6.6.5 (Continuità dell'operatore di punto fisso)**.** *Sia D un dominio, per il teorema del punto fisso di Tarski (6.6.6) ciascuna funzione continua $f \in (D \to D)$ ha minimo punto fisso $fix(f) \in D$. Allora, la funzione*

$$fix : (D \to D) \to D$$

è continua. ◊

Dimostrazione. Bisogna prima dimostrare che $fix : (D \to D) \to D$ è una funzione monotona. Si supponga che $f_1 \sqsubseteq f_2$ sia nel dominio funzione $D \to D$. Bisogna dimostrare che $fix(f_1) \sqsubseteq fix(f_2)$. Ma:

$$\begin{array}{ll} f_(fix(f_2)) \sqsubseteq f_2(fix(f_2)) & (f_1 \sqsubseteq f_2) \\ \sqsubseteq fix(f_2) & ((6.4) \text{ per } fix(f_2)) \end{array}$$

Quindi $fix(f_2)$ è un punto prefisso per f_1 e da (6.5) (per $fix(f_1)$), segue $fix(f_1) \sqsubseteq fix(f_2)$.

Per quanto riguarda il mantenimento dei minimi comuni maggioranti delle catene, si supponga di avere $f_0 \sqsubseteq f_1 \sqsubseteq f_2 \sqsubseteq \ldots$ in $D \to D$. Per l'osservazione (6.1.5) bisogna dimostrare che

$$fix(\bigsqcup_{n\geq 0} f_n) \sqsubseteq \bigsqcup_{n\geq 0} fix(f_n)$$

Per la proprietà (6.5) è sufficiente mostrare che $\sqcup_{n\geq 0} fix(f_n)$ è un punto prefisso per la funzione $\sqcup_{n\geq 0} f_n$. Ciò è vero poiché:

$(\bigsqcup_{m\geq 0} f_m)(\bigsqcup_{n\geq 0} fix(f_n)) = \bigsqcup_{m\geq 0} f_m(\bigsqcup_{n\geq 0} fix(f_n))$ (minimi comuni maggioranti delle funzioni calcolati argomento per argomento)

$= \bigsqcup_{m\geq 0}\bigsqcup_{n\geq 0} f_m(fix(f_n))$ (continuità di ogni f_m)

$= \bigsqcup_{k\geq 0} f_k(fix(f_k))$ (lemma (6.3.4))

$\sqsubseteq \bigsqcup_{k\geq 0} fix(f_k)$ ((6.4) per ogni f_k)

□

6.6.1 Teorema del punto fisso di Tarski

Teorema 6.6.6 (Punto fisso di Tarski)**.** *Sia $f : D \to D$ una funzione continua su un dominio D. Allora*

- *f ha un minimo punto prefisso dato da*

$$fix\,(f) = \bigsqcup_{n\geq 0} f^n\,(\bot)$$

- *$fix\,(f)$ è un punto fisso di f i.e. soddisfa $f\,(fix\,(f)) = fix\,(f)$ e quindi è il minimo punto fisso di f.*

◊

Questo teorema fornisce il risultato chiave sulle funzioni continue su domini e consente di fornire la semantica denotazionale di programmi che hanno caratteristiche ricorsive.

Definizione 6.6.7 ($f^n\,(\bot)$)**.** *Si definisce $f^n\,(\bot)$ come:*

$$\begin{cases} f^0\,(\bot) & \stackrel{def}{=} & \bot \\ f^{n+1}\,(\bot) & \stackrel{def}{=} & f\,(f^n\,(\bot)) \end{cases} \tag{6.6}$$

◊

Poiché $\forall d \in D \;\; \bot \sqsubseteq d$:

$$f^0\,(\bot) = \bot \sqsubseteq f^1\,(\bot)$$

Per la monotonia della funzione f:

$$f^n(\bot) \sqsubseteq f^{n+1}(\bot) \Rightarrow$$
$$f^{n+1}(\bot) = f(f^n(\bot)) \sqsubseteq f\left(f^{n+1}(\bot)\right) = f^{n+2}(\bot)$$

Quindi, per induzione su $n \in \mathbb{N}$, si ha che $\forall n \in \mathbb{N}\ f^n(\bot) \sqsubseteq f^{n+1}(\bot)$. In altre parole, gli elementi $f^n(\bot)$ formano una catena nell'insieme D.

Dimostrazione (Teorema del Punto Fisso di Tarski).

$$f(fix(f)) = f\left(\bigsqcup_{n\geq 0} f^n(\bot)\right)$$

$$= \bigsqcup_{n\geq 0} f(f^n(\bot)) \qquad \text{(per la continuità di } f\text{)}$$

$$= \bigsqcup_{n\geq 0} f^{n+1}(\bot) \qquad \text{(per la proprietà (6.6))}$$

$$= \bigsqcup_{n\geq 0} f^n(\bot) \qquad \text{(scarto di un \# finito di elementi all'inizio delle catena)}$$

$$= fix(f)$$

Quindi, $fix(f)$ è un punto fisso per f ed in particolare soddisfa la proprietà (6.4). Per verificare la proprietà (6.5), supponendo che per $d \in D$ valga $f(d) \sqsubseteq d$, poiché $\bot$ è il più piccolo elemento nell'insieme D,

$$f^0(\bot) = \bot \sqsubseteq d$$

Per la monotonia di f e per $f(d) \sqsubseteq d$:

$$f^n(\bot) \sqsubseteq d \quad \Rightarrow \quad f^{n+1}(\bot) = f(f^n(\bot)) \sqsubseteq f(d) \sqsubseteq d$$

Quindi, per induzione su $n \in \mathbb{N}$ si ha che $\forall n \in \mathbb{N}\ f^n(\bot) \sqsubseteq d$. Allora d è un massimo della catena e quindi:

$$fix(f) = \bigsqcup_{n\geq 0} f^n(\bot) \sqsubseteq d$$

come richiesto dalla proprietà (6.5). □

Osservazione 6.6.8. Si consideri il minimo $\bot$ e si consideri $f(\bot)$. Poiché $\bot$ è il minimo vale $\bot \leq f(\bot)$.

Se $\perp = f(\perp)$ si è trovato un punto fisso. Altrimenti si riprova considerando f^2: applicando f alla disuguaglianza $\perp \leq f(\perp)$ si ottiene $f(\perp) \leq f^2(\perp)$ (si osservi che questo passaggio è corretto poiché f è continua, monotona e $\perp \leq f(\perp)$).

Se $f(\perp) = f^2(\perp)$ ci si ferma, altrimenti iterando il procedimento si ottiene $f^2(\perp) \leq f^3(\perp)$.

Si ottiene così la catena di disuguaglianze:

$$\perp \leq f(\perp) \leq f^2(\perp) \leq f^3(\perp) \leq \dots \tag{6.7}$$

La (6.7) potrebbe essere una catena infinita, oppure potrebbe arrestarsi ad un certo punto, individuando così un punto fisso.

Qualora la catena fosse infinita, essendo totalmente ordinata in un CPO avrebbe comunque un minimo comune maggiorante.

Si dimostra ora che il minimo comune maggiorante di (6.7) i.e. $\bigsqcup_{k\in\mathbb{N}} f^k(\perp)$, è il punto fisso cercato, ed è proprio il minimo punto fisso.

($\bigsqcup_{k\in\mathbb{N}} f^k(\perp)$ è un punto fisso)

Applicando f a $\bigsqcup_{k\in\mathbb{N}} f^k(\perp)$ si ottiene:

$$f\left(\bigsqcup_{k\in\mathbb{N}} f^k(\perp)\right) = \bigsqcup_{k\in\mathbb{N}} f^{k+1}(\perp).$$

Cerchiamo ora di capire di quale catena $\bigsqcup_{k\in\mathbb{N}} f^{k+1}(\perp)$ è il minimo comune maggiorante.

Per k=0, $\bigsqcup_{k\in\mathbb{N}} f^{k+1}(\perp) = f(\perp)$.

Per k=1, $\bigsqcup_{k\in\mathbb{N}} f^{k+1}(\perp) = f^2(\perp)$.

Iterando il procedimento si ottiene la seguente catena:

$$f(\perp) \leq f^2(\perp) \leq f^3(\perp) \leq f^4(\perp) \leq \dots \tag{6.8}$$

Si osservi che la catena (6.8) risulta sfalsata di un elemento rispetto alla catena (6.7), quindi differiscono l'una dall'altra per il primo elemento. Tuttavia, per il lemma (6.1.8) (dell'intreccio di catene), il minimo comune maggiorante di (6.8) coincide con il minimo comune maggiorante di (6.7). Infatti, per ogni elemento di (6.7) c'è un elemento di (6.8) che lo maggiora e viceversa.

Quindi si è dimostrato che

$$\bigsqcup_{k\in\mathbb{N}} f^k(\bot) = \bigsqcup_{k\in\mathbb{N}} f^{k+1}(\bot) = f\left(\bigsqcup_{k\in\mathbb{N}} f^k(\bot)\right)$$

è un punto fisso.

($\bigsqcup_{k\in\mathbb{N}} f^k(\bot)$ è il minimo punto fisso)

Si procede dimostrando che dato un qualunque altro punto fisso questo risulta essere non inferiore.

Si supponga che esista un altro punto fisso $\overline{x}$ e si consideri la catena:

$$\overline{x} \leq f(\overline{x}) \leq f^2(\overline{x}) \leq f^3(\overline{x}) \leq \ldots \tag{6.9}$$

Poiché $\overline{x}$ è un punto fisso si ha: $\overline{x} = f(\overline{x})$, $\overline{x} = f^2(\overline{x})$, $\overline{x} = f^3(\overline{x})$,...

Analizziamo ora in che relazione sono gli elementi della catena (6.9) con gli elementi della catena (6.7):

- $\bot \leq \overline{x}$ poiché $\bot$ è il minimo
- $f(\bot) \leq f(\overline{x})$ (f monotona)
- $f^2(\bot) \leq f^2(\overline{x})$
- $f^i(\bot) \leq f^i(\overline{x})$ (iterando)

La catena (6.9) è una maggiorante della catena (6.7) (ogni elemento di (6.9) è maggiore di ogni elemento di (6.7)), e quindi anche il minimo comune maggiorante di (6.9) sarà maggiore o uguale al minimo comune maggiorante di (6.7).

Si osservi che, essendo la catena (6.9) costante, il minimo comune maggiorante di (6.9) è $\overline{x}$, che è maggiore o uguale di ogni elemento di (6.7) e quindi anche del suo minimo comune maggiorante, che è il punto fisso di (6.7), cioè:

$$\overline{x} \geq \bigsqcup_{k\in\mathbb{N}} f^k(\bot).$$

Quindi $\bigsqcup_{k\in\mathbb{N}} f^k(\bot)$ è il minimo punto fisso. ◊

Esempio 6.6.9 (Numeri naturali)**.** Si consideri l'insieme dei numeri naturali $\mathbb{N}$ e la funzione successore: non esiste il punto fisso poiché il successore di un numero n è sempre diverso da n. ⊠

Capitolo 7

Ancora sull'algebra dei linguaggi

Si ricorda che $(\mathcal{P}(A^*), \subseteq)$ è un insieme parzialmente ordinato in cui $\emptyset$ è il minimo e A^* è il massimo.

Quindi si osservi che $\left(\mathcal{P}(A^*), \cap, \cup, ()'\right)$ è un'algebra di Boole, ossia è un reticolo distributivo complementato.

Esempio 7.0.10 (Anello)**.** Ci si potrebbe chiedere se, considerando $\cup$ come somma e $\cdot$ come prodotto, $\mathcal{P}(A^*)$ risulti essere un anello[1]. Infatti la somma $\cup$ è associativa, commutativa ed esiste l'elemento neutro $\emptyset$.

Sono verificate le leggi distributive, ossia $\forall L, M, N \in \mathcal{P}(A^*)$ risulta:

$$L \cdot (M \cup N) = (L \cdot M) \cup (L \cdot N)$$

dove $L \cdot (M \cup N)$ per definizione di unione è la concatenazione di una parola di L con una parola di M oppure con una di N. Dunque è uguale a $(L \cdot M) \cup (L \cdot N)$ che è una parola di $L \cdot M$ oppure di $L \cdot N$. Analogamente si dimostra che $(M \cup N) \cdot L = (M \cdot L) \cup (N \cdot L)$.

Ciò che manca a questa struttura per essere un anello è l'esistenza dell'opposto rispetto alla somma, che nel nostro caso è l'operazione $\cup$: $\forall L \in \mathcal{P}(A^*) \setminus \{\emptyset\}$ non esiste alcun elemento di $\mathcal{P}(A^*)$ che sommato ad L dia come risultato $\emptyset$.

In conclusione $(\mathcal{P}(A^*), \cup, \cdot, 0, \{\varepsilon\})$ è un semianello ordinato. ⊠

Da tutto ciò segue che, se A è un alfabeto, $\mathcal{P}(A^*)$ ha una struttura algebrica ricchissima:

$$(\mathcal{P}(A^*), \subseteq, \cup, \cap, \emptyset, A^*, \cdot, \{\varepsilon\})$$

[1]Per le definizioni di anello e semianello si consulti la tabella 7.1

Tabella 7.1: Definizioni di anello e semianello

Si ricorda che un *anello* è un insieme A dotato di due operazioni binarie $+$ e $\cdot$, tali che:

- $(A, +)$ è un gruppo abeliano con elemento neutro 0:
  - $(a + b) + c = a + (b + c)$
  - $a + b = b + a$
  - $0 + a = a + 0 = a$
  - $\forall a \ \exists(-a)$ tale che $a + \ -a = -a + a = 0$
- $(A, \cdot)$ è un monoide con elemento neutro 1:
  - $1 \cdot a = a \cdot 1 = a$
  - $(a \cdot b) \cdot c = a \cdot (b \cdot c)$
- La moltiplicazione è distributiva rispetto alla somma:
  - $a \cdot (b + c) = (a \cdot b) + (a \cdot c)$
  - $(a + b) \cdot c = (a \cdot c) + (b \cdot c)$

(le relazioni devono valere per ogni a, b e c in A).

Un *semianello* è un insieme A dotato di due operazioni binarie $+$ e $\cdot$ tali che:

- $(A, +)$ è un monoide commutativo con elemento neutro 0:
  - $(a + b) + c = a + (b + c)$
  - $0 + a = a + 0 = a$
  - $a + b = b + a$
- $(A, \cdot)$ è un monoide con elemento neutro 1:
  - $(a \cdot b) \cdot c = a \cdot (b \cdot c)$
  - $1 \cdot a = a \cdot 1 = a$
- La moltiplicazione è distributiva rispetto alla somma:
  - $a \cdot (b + c) = (a \cdot b) + (a \cdot c)$
  - $(a + b) \cdot c = (a \cdot c) + (b \cdot c)$
- 0 annulla A:
  - $0 \cdot a = a \cdot 0 = 0$

A questo punto possiamo considerqare una nuova operazione che ha, però, carattere infinitario.

Definizione 7.0.11 (Star di Kleene)**.** *Sia L un insieme. Si denota con L^* l'insieme ottenuto come "iterazione" di L i.e. l'insieme i cui elementi sono tutte le concatenazioni finite di L:*

- $L^0 = \{\varepsilon\}$
- $L^{n+1} = L \cdot L^n$
- $L^* = \bigcup_{n\in\mathbb{N}} L^n$

$\Diamond$

Esempio 7.0.12 (Star di Kleene)**.** Sia $L = \{a\}$, risulta $L^* = \{\varepsilon, a, aa, aaa, aaaa, \ldots.\} = \{a^n\}_{n\in\mathbb{N}}$.

$\boxtimes$

L'operazione star di Kleene è La grammatica

$$\left\{ \; X = aX \mid \varepsilon$$

"genera", attraverso la tecnica del punto fisso, questo linguaggio. Una grammatica libera da conteso è un operatore continuo e può essere pensata come un sistema di equazioni in n-variabili, se n è il numero dei simboli non terminali. La minima soluzione del sistema sarà allora il linguaggio associato.

Esempio 7.0.13. Si consideri un'equazione del tipo[2]:

$$x = ax + b$$

tale che:

$$\mathcal{P}(A^*) \overset{f=a\cdot\_+b}{\longrightarrow} \mathcal{P}(A^*)$$

Bisogna dimostrare che questa funzione è continua, così da poter applicare il teorema del punto fisso e trovare il minimo punto fisso.

Per verificare la monotonia, si consideri un Y tale che:

$$X \subseteq Y \;\Rightarrow\; aX + b \;\subseteq\; aY + b$$

[2]Potrebbe essere anche del tipo $x = (a + a')\, x + b$ oppure $x = (a + a')\, x + (b + b')$, dato che in ogni caso ci si può sempre ricondurre alla prima equazione poiché vale la proprietà distributiva e si può raccogliere a fattor comune.

Tabella 7.2: Stephen Cole Kleene

Tratto da [Wik06].

Stephen Cole Kleene (5 Gennaio 1909 - 25 Gennaio 1994) fu un matematico americano che lavorò all'*università di Wisconsin-Madison* dove predispose le fondamenta dell'informatica teorica. Kleene è anche noto per la fondazione del ramo della logica matematica conosciuta come *teoria della ricorsione* insieme con *Alonzo Church*, *Kurt Gödel*, *Alan Turing* ed altri; e per l'aver inventato le espressioni regolari. Fornendo metodi per determinare quali problemi sono risolvibili, il lavoro di Kleene portò allo studio di quali funzioni fossero calcolabili. Ricordiamo inoltre, l'*algebra di Kleene*, la *star di Kleene*, il *teorema di ricorsione di Kleene* ed il *teorema di punto fisso di Kleene*. Ha anche contribuito all'intuizionismo matematico fondato da *Luitzen Egbertus Jan Brouwer*.

Kleene si pronuncia ['klej.ni] (IPA). ['kli.ni] e [klin] sono pronunce errate estremamente comuni.

La posizione di Kleene nella logica matematica è riflessa nel proverbio "Kleeneliness is next to Gödeliness" tra logici (un gioco di parole su "Cleanliness is next to godliness").

Kleene naque a Hartford, Connecticut, USA. Ricevette il bachelor of arts degree dall'*Amherst College* nel 1930. Dal 1930 al 1935, fu un graduate student e un assistente ricercatore all'*università di Princeton*, dove conseguì il dottorato in matematica nel 1934, supervisionato da Alonzo Church, per una tesi intitolata *A Theory of Positive Integers in Formal Logic*. Nel 1935, si unì al dipartimento di matematica di *UW-Madison* come istruttore. Divenne assistente professore nel 1937.

Dal 1939 al 1940, fu un visiting scholar all'*istituto per gli studi avanzati di Princeton*, dove pose le basi per la *teoria delle funzioni ricorsive*, un'area che fu il suo interesse di ricerca per l'arco della sua vita. Nel 1941 tornò ad Amherst come professore associato di matematica.

Durante la seconda guerra mondiale, Kleene fu un comandante luogotenente nella *marina degli Stati Uniti*. Fu istruttore di navigazione alla *U.S. Naval Reserve's Midshipmen's School* a New York, e dopo un direttore di progetto presso il *laboratorio di ricerca navale* a Washington, D.C.

Nel 1946, tornò in Wisconsin, diventando professore ordinario nel 1948. Ebbe la cattedra di matematica e informatica nel 1962 e nel 1963, e fu decano del *collegio di Lettere e Scienza* dal 1969 al 1974. Nel 1964 fu nominato professore di matematica *Cyrus C. MacDuffee*. Si ritirò nel 1979.

Scalatore, Kleene ebbe forti interessi in natura e ambiente e fu attivo in molte cause di conservazione. Fu presidente della *associazione di Logica Simbolica* dal 1956 al 1958 e della *unione internazionale della storia e della filosofia della scienza* nel 1961. Morì in Madison, Wisconsin.

Quindi, vale la monotonia.

Per la continuità (6.1.2):

$$X_1 \subseteq X_2 \subseteq X_3 \subseteq X_4 \subseteq \ldots \subseteq \cup_i X_i$$

$$\cup_i a(X_i) + b \quad \stackrel{?}{=} \quad a(\cup_i X_i) + b \tag{7.1}$$

Al membro destro di (7.1) si ha $a\, w_i$ oppure b, al membro sinistro si ha $a\, w_i$ oppure b: essendo uguali la funzione è monotona e continua.

Applicando quanto ottenuto:
$x = ax + b$
$x_0 = 0$
$x_1 = ax_0 + b \Rightarrow x = b$
$x_2 = ax_1 + b \Rightarrow x = ab + b$
$x_3 = ax_2 + b \Rightarrow x = a\,(ab + b) + b = a^2b + ab + b$
...

Iterando il procedimento si ottiene che la minima soluzione di questa equazione (i.e. il minimo punto fisso) è:

$$x = a^*b$$

Riassumendo:

$$\emptyset \subseteq f\,(\emptyset) = \{b\} \subseteq f\,(f\,(\emptyset)) = \{ab + b\} \subseteq f\,(f\,(f\,(\emptyset))) = \{a\,(ab + b) + b\} = a^2b + ab + b \subseteq \ldots \subseteq a^*b$$

Vedendo la soluzione dell'equazione da un punto di vista algebrico:

$$x = \frac{1}{1-a}b$$

Per $a < 1$:

$$\frac{1}{1-a} = 1 + a + a^2 + a^3 + \ldots$$

Ci sono quindi delle somiglianze nei due diversi modi di interpretare l'equazione.

⊠

Esempio 7.0.14. Si consideri una funzione del tipo:

$$x = axb + ab$$

Applicando il metodo per sostituzione visto in precedenza:
$x_0 = 0$
$x_1 = a \cdot 0 \cdot b + ab = ab$
$x_2 = a \cdot x_1 \cdot b + ab = aabb + ab = a^2b^2 + ab$

$x_3 = ax_2b + ab = a^3b^3 + a^2b^2 + ab$
$x_4 = \ldots$

Questo procedimento va avanti all'infinito, ed ogni linguaggio ottenuto contiene il precedente:

$$\emptyset \subseteq \{ab\} \subseteq \left\{a^2b^2, ab\right\} \subseteq \left\{a^3b^3, a^2b^2, ab\right\} \subseteq \ldots \subseteq \{a^nb^n\}$$

Quindi a^nb^n è il minimo punto fisso di questa catena. ⊠

Teorema 7.0.15 (Minimo punto fisso di una grammatica)**.** *(Kleene) Sia G una grammatica libera da contesto con n simboli non terminali e con funzione associata f. f è una funzione continua da $P(A*)^n$ pensato come dominio, in sé. Allora, il minimo punto fisso di f è il linguaggio libero da contesto $L(G)$ generato da G.* ◊

Esempio 7.0.16.

$$\begin{cases} X = aX + bY + 1 \\ Y = cX + dY + 1 \end{cases}$$

La corrispondente grammatica regolare è:

$$X \to aX \mid bY \mid \varepsilon$$
$$Y \to cX \mid dY \mid \varepsilon$$

Per sostituzione:

1. $X_0 = 0;\ Y_0 = 0;$

$$\begin{cases} X_1 = 1 = \varepsilon \\ Y_1 = 1 = \varepsilon \end{cases}$$

2. $X_1 = 1;\ Y_1 = 1;$

$$\begin{cases} X = a + b + 1 \\ Y = c + d + 1 \end{cases}$$

3. $X_2 = a + b + 1;\ Y_2 = c + d + 1;$

$$\begin{cases} X_2 = a\,(a + b + 1) + b\,(c + d + 1) + 1 \\ Y_2 = c\,(a + b + 1) + d\,(c + d + 1) + 1 \end{cases}$$

4. $\ldots$

⊠

Abbiamo visto l'esempio con una grammatica lineare destra, ma il metodo, grazie al teorema (7.0.15), è applicabile anche per le grammatiche libere da contesto, in cui a sinistra di ogni produzione c'è un simbolo non terminale.

Esempio 7.0.17.

$$\begin{array}{ll} X \to aXa \mid b \\ Y \to Ya^2 \mid bXY \end{array} \quad \Rightarrow \quad \begin{array}{l} X = aXa + b \\ Y = Ya^2 + bXY \end{array}$$

Anche in questo caso è possibile trovare il minimo punto fisso:

1. $X_0 = 0;$
   $Y_0 = 0;$
2. $X_1 = b;$
   $Y_1 = 0;$
3. $X_2 = aba + b;$
   $Y_2 = 0;$
4. $X_3 = aabaa + aba + b;$
   $Y_3 = 0;$
5. ...

n. $X_n = a^n b a^n;$
   $Y_n = 0;$
   $\Rightarrow$ minimo punto fisso

⊠

Esempio 7.0.18.

$$\begin{cases} X = aXYa + b \\ Y = Xb + 1 \end{cases}$$

1. $X_0 = 0;$
   $Y_0 = 0;$
2. $X_1 = b;$
   $Y_1 = 1;$
3. $X_2 = aba + b;$
   $Y_2 = bb + 1;$
4. $X_3 = a\,(aba + b) \cdot (bb + 1) \cdot a + b;$
   $Y_3 = (aba + b) \cdot b + 1;$
5. ...

⊠

Nelle grammatiche lineari destre è sempre possibile trovare il minimo punto fisso i.e. la soluzione dell'equazione utilizzando l'operatore $()^*$ di Kleene; infatti possiamo risolvere ogni sistema di equazioni lineari usando la cosiddetta Regola di Arden

$$X = LX + M \quad \Rightarrow \quad X = L^*M$$

che afferma in sostanza che L^*M è il minimo punto fisso dell'operatore continuo $L - +M$.

Esempio 7.0.19. Riprendiamo l'esempio (7.0.16):

$$\begin{cases} X = aX + bY + 1 \\ Y = cX + dY + 1 \end{cases}$$

Procedendo come segue:

$Y = cX + dY + 1 \quad \Rightarrow \quad Y = dY + (cX + 1)$ (per la proprietà associativa)

si ottiene la soluzione:

$$Y = d^* \cdot (cX + 1)$$

Ora, sostituendo questa soluzione nella prima equazione:

$$X = aX + bd^* \cdot (cX + 1) + 1$$

Per la proprietà distributiva destra:

$$X = aX + bd^*cX + bd^* + 1$$

Per la proprietà distributiva sinistra e per l'associatività:

$$X = \underbrace{(a + bd^*c)}_{a} X + \underbrace{(bd^* + 1)}_{b}$$

Ci si è ricondotti al caso di un esempio visto in precedenza (7.0.13) in cui l'equazione era $x = ax + b$ e la soluzione (o minimo punto fisso) era $x = a^*b$, quindi:

$$X = (a + bd^*c)^* \cdot (bd^* + 1)$$
$$Y = d^* \left(c \left(a + bd^*c\right)^* \left(bd^* + 1\right) + 1\right)$$

che rappresenta il minimo punto fisso. ⊠

Capitolo 8

Principio di induzione di Scott sui punti fissi

Ricordiamo che il minimo punto fisso di una funzione continua $f : D \to D$ su un dominio $(D, \sqsubset)$ può essere espresso come il minimo comune maggiorante di una catena costruita applicando ripetutamente la funzione f partendo dall'elemento minimo $\bot \in D$: $\quad fix\,(f) = \bigsqcup_{n \geq 0} f^n\,(\bot)$ (si veda il teorema 6.6.6).

Definizione 8.0.20 (CPO chiuso per catene)**.** *Sia $(D, \sqsubseteq)$ un CPO, un sottoinsieme $S \subseteq D$ è* chiuso per catene *se e solo se per ogni catena $d_0 \sqsubseteq d_1 \sqsubseteq d_2 \sqsubseteq d_3 \sqsubseteq \ldots \in D$ vale:*

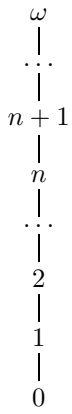
$$(\forall n \geq 0 \;\; d_n \in S) \;\Rightarrow\; \left(\bigsqcup_{n \geq 0} d_n \right) \in S$$

$\Diamond$

Definizione 8.0.21 (Insieme ammissibile)**.** *Sia D un dominio, $S \subseteq D$ è* ammissibile *se e solo se è un sottoinsieme di D chiuso per catene e $\bot \in S$.*
$\Diamond$

Definizione 8.0.22. *Una proprietà $\Phi\,(d)$ degli elementi $d \in D$ è detta* chiusa per catene/ammissibile *se e solo se $\{d \in D \mid \Phi\,(d)\}$ è un sottoinsieme di D* chiuso per catene/ammissibile*.* $\Diamond$

Esempio 8.0.23. Si consideri il dominio dei numeri naturali "verticali" Ω:



Si ha che:

- ogni sottoinsieme finito di Ω è chiuso per catene
- il sottoinsieme $\{0, 2, 4, 6, \ldots\}$ non è un sottoinsieme di Ω chiuso per catene
- il sottoinsieme $\{0, 2, 4, 6, \ldots\} \cup \{\omega\}$ è un sottoinsieme di Ω chiuso per catene ed ammissibile

$\boxtimes$

Le equazioni di punto fisso si presentano molto spesso quando si fornisce una semantica denotazionale ai linguaggi con caratteristiche ricorsive. Dana Scott, negli anni '60, creò una teoria matematica chiamata *teoria dei dominî*, sviluppando un metodo con cui sarebbe stato possibile non solo trovare sempre la soluzione delle equazioni di punto fisso per la semantica denotazionale, ma anche trovare soluzioni che sarebbero state minimali, in modo da avere una corrispondenza esatta tra la semantica denotazionale e quella operazionale.

Esempio 8.0.24 (Proprietà del punto fisso di $[\![\texttt{while}\ B\ \texttt{do}\ C]\!]$)**.** Sia:

$$[\![\texttt{while}\ B\ \texttt{do}\ C]\!] = f_{[\![B]\!],[\![C]\!]}\left([\![\texttt{while}\ B\ \texttt{do}\ C]\!]\right)$$

dove per ogni $b : State \to \{true, false\}$ e $c, w : State \to State$ si definisce:

$$f_{b,c}(w) = \lambda s \in State.if\left(b(s), w(c(s)), s\right) \tag{8.1}$$

$\boxtimes$

Tabella 8.1: Dana S. Scott

Da [Wik06].
Dana S. Scott (nato nel 1932) è un professore emerito di informatica all'*Hillman University*, filosofia, e logica matematica alla *Carnegie Mellon University*; ora si è ritirato e vive a Berkeley, California. La sua carriera di ricerca ha spaziato tra informatica, matematica e filosofia, ed è stato caratterizzato dal matrimonio tra un'attenzione per la spiegazione dei concetti fondamentali in maniera informale, con la coltivazione di problemi matematici difficili basati su questi concetti. Il suo lavoro in *teoria degli automi* gli fece conseguire l'*ACM Turing Award* nel 1976, mentre la sua collaborazione con *Christopher Strachey* negli anni '70 posero le basi per i moderni approcci alla *semantica dei linguaggi di programmazione*. Ha lavorato anche sulla *logica modale*, *topologia* and *teoria delle categorie*. È editor-in-chief della nuova rivista *Logical Methods in Computer Science*.

L'idea è di considerare un ordine parziale tra gli oggetti matematici usati come denotazioni, in cui si esprime che un oggetto "è approssimato per" o "possiede più informazioni di" o "è più definito di". Allora, la soluzione minima di un'equazione di punto fisso può essere costruita come limite di una catena crescente di approssimazioni della soluzione.

Esempio 8.0.25. Si consideri:

$$\texttt{while}\ X > 0\ \texttt{do}\ (\ Y := X * Y\ ;\ X := X - 1\ ) \tag{8.2}$$

dove X e Y sono due distinte locazioni di memoria (variabili di tipo intero). In questo caso uno stato è una coppia di interi (x, y) che tiene traccia del contenuto corrente di X e Y (rispettivamente). Quindi, $State = \mathbb{Z} \times \mathbb{Z}$, (o, indifferentemente, $State = \mathbb{N} \times \mathbb{N}$).

Proviamo a definire la denotazione di (8.2) come una funzione parziale $w : (\mathbb{Z} \times \mathbb{Z}) \rightharpoonup (\mathbb{Z} \times \mathbb{Z})$; la denotazione dovrà essere una soluzione dell'equazione (8.1).

Per l'espressione booleana $B = (X > 0)$ ed il comando $C = (Y := X * Y; X := X - 1)$, la funzione $f_{[\![B]\!],[\![C]\!]}$ coincide con la funzione f che ora vedremo (8.3).

Come detto, si hanno stati definiti come coppie di interi:

$$State \stackrel{def}{=} \mathbb{Z} \times \mathbb{Z}$$

e funzioni parziali:

$$D \stackrel{def}{=} State \rightharpoonup State$$

Per $[\![\texttt{while}\ X > 0\ \texttt{do}\ Y := X * Y; X := X - 1]\!] \in D$ cercheremo la soluzione minima per $w = f(w)$, dove $f : D \to D$ è definita da:

$$f(w)(x,y) = \begin{cases} (x,y) & \text{se } x \leq 0 \\ w(x-1, x*y) & \text{se } x > 0 \end{cases} \tag{8.3}$$

Consideriamo l'ordine parziale $\sqsubseteq$ tra elementi di D:

$w \sqsubseteq w' \Leftrightarrow$

$\forall (x,y) \in State$

se w è definito per (x,y) allora lo è anche w' ed inoltre $w(x,y) = w'(x,y)$.

Si ricorda, inoltre, che il più piccolo elemento $\perp \in D$ rispetto a $\sqsubseteq$ è la funzione parziale totalmente indefinita e vale: $\forall w \in D \ \perp \sqsubseteq w$.

Iniziamo con l'elemento minimo $\perp$, applichiamo ripetutamente f per costruire una sequenza di funzioni parziali $w_0, w_1, w_2, \ldots$:

$$\begin{cases} w_0 & \stackrel{def}{=} & \perp \\ w_{n+1} & \stackrel{def}{=} & f(w_n) \end{cases}$$

Utilizzando la definizione di f in (8.3) si ottiene:

$$w_1(x,y) = f(\perp)(x,y) = \begin{cases} (x,y) & \text{se } x \leq 0 \\ indefinita\ (\perp) & \text{se } x \geq 1 \end{cases}$$

$$w_2(x,y) = f(w_1)(x,y) = \begin{cases} (x,y) & \text{se } x \leq 0 \\ (0,y) & \text{se } x = 1 \\ indefinita\ (\perp) & \text{se } x \geq 2 \end{cases}$$

$$w_3(x,y) = f(w_2)(x,y) = \begin{cases} (x,y) & \text{se } x \leq 0 \\ (0,y) & \text{se } x = 1 \\ (0, 1*2*y) & \text{se } x = 2 \\ indefinita\ (\perp) & \text{se } x \geq 3 \end{cases}$$

$$w_4(x,y) = f(w_3)(x,y) = \begin{cases} (x,y) & \text{se } x \leq 0 \\ (0,y) & \text{se } x = 1 \\ (0, 1*2*y) & \text{se } x = 2 \\ (0, 1*2*3*y) & \text{se } x = 3 \\ indefinita\ (\perp) & \text{se } x \geq 4 \end{cases}$$

$\ldots$

In generale:

$$w_n\,(x,y) = f\,(w_{n-1})\,(x,y) = \begin{cases} (x,y) & \text{se } x \leq 0 \\ (0,(x!)*y) & \text{se } 0 < x < n \\ indefinita\ (\perp) & \text{se } x \geq 0 \end{cases}$$

Si ottiene così una sequenza crescente di funzioni parziali:

$$w_0 \sqsubseteq w_1 \sqsubseteq w_2 \sqsubseteq \ldots \sqsubseteq w_n \sqsubseteq \ldots$$

definite su insiemi sempre più grandi di stati (x, y).

L'unione di tutte queste funzioni parziali è l'elemento $w_\infty \in D$:

$$w_\infty\,(x,y) = \begin{cases} (x,y) & \text{se } x \leq 0 \\ (0,(x!)*y) & \text{se } x > 0 \end{cases}$$

Si osservi che w_∞ è un punto fisso della funzione f poiché per ogni (x, y) si ha (per definizione di f):

$$f\,(w_\infty)\,(x,y) = \begin{cases} (x,y) & \text{se } x \leq 0 \\ w_\infty\,(x-1, x*y) & \text{se } x > 0 \end{cases}$$

$$= \begin{cases} (x,y) & \text{se } x \leq 0 \\ (0,1*y) & \text{se } x = 1 \\ (0,(x-1)!*x*y) & \text{se } x > 1 \end{cases} \qquad \text{(definizione di } w_\infty\text{)}$$

$$= w_\infty\,(x,y)$$

Si può dimostrare che w_∞ è il minimo punto fisso della funzione f, nel senso che:

$$\forall w \in D \;\; w = f\,(w) \;\Rightarrow\; w_\infty \sqsubseteq w$$

Questo minimo punto fisso può essere considerato come la denotazione di:

$$\texttt{while}\ X > 0\ \texttt{do}\ (\ Y := X * Y\ ;\ X := X - 1\ )$$

La sua costruzione è un'istanza del teorema del punto fisso di Tarski (6.6.6).

⊠

Teorema 8.0.26 (Principio di induzione di Scott)**.** *Sia $f : D \to D$ una funzione continua su un dominio D. Per ogni sottoinsieme ammissibile $S \subseteq D$, per provare che:*

$$fix\,(f) \in S$$

è sufficiente dimostrare che:

$$\forall d \in D \;\; d \in S \Rightarrow f\,(d) \in S$$

◊

Osservazione 8.0.27 (IP di Scott). Il principio di induzione di Scott è una forma alternativa del teorema di Tarski. In generale, S non coincide con D. $\Diamond$

Esempio 8.0.28. Sia D un dominio e sia $f : (D \times (D \times D)) \to D$ una funzione continua. Sia $g : (D \times D) \to (D \times D)$ una funzione continua così definita:

$$g\,(d_1, d_2) \stackrel{def}{=} (f\,(d_1, (d_1, d_2)), f\,(d_1, (d_2, d_2))) \qquad (d_1, d_2 \in D)$$

Allora $u_1 = u_2$, dove $(u_1, u_2) \stackrel{def}{=} fix\,(g)$. $\boxtimes$

Dimostrazione. Bisogna dimostrare che $fix\,(g) \in \Delta$ con Δ così definito:

$$\Delta \stackrel{def}{=} \{(d_1, d_2) \in D \times D \mid d_1 = d_2\}$$

Si osservi che Δ è un sottoinsieme ammissibile del dominio prodotto $D \times D$. Applicando il teorema (8.0.26) bisogna verificare che:

$$\forall\,(d_1, d_2) \in D \times D \quad (d_1, d_2) \in \Delta \Rightarrow g\,(d_1, d_2) \in \Delta$$

o, equivalentemente, che:

$$\forall\,(d_1, d_2) \in D \times D \quad d_1 = d_2 \Rightarrow f\,(d_1, d_1, d_2) = f\,(d_1, d_2, d_2)$$

(che è chiaramente vero). $\square$

Esempio 8.0.29. Sia $f : D \to D$ la funzione continua definita in (8.3), in cui il minimo punto fisso è la denotazione del comando:

$$\texttt{while}\ X > 0\ \texttt{do}\ (\ Y := X * Y\ ;\ X := X - 1\ )$$

Useremo il principio di induzione di Scott per provare che:

$$\forall x, y \geq 0 \qquad fix\,(f)\,(x, y) \neq \bot \quad \Rightarrow \quad fix\,(f)\,(x, y) = (0, (x!) * y) \qquad (8.4)$$

dove per $w \in D = (\mathbb{Z} \times \mathbb{Z}) \rightharpoonup (\mathbb{Z} \times \mathbb{Z})$ si usa $w\,(x, y) \neq \bot$ per indicare che la funzione parziale w è definita in (x, y). In altre parole, si può identificare D con il dominio delle funzioni (continue) dal CPO discreto $(\mathbb{Z} \times \mathbb{Z})$ al dominio piatto $(\mathbb{Z} \times \mathbb{Z})_\bot$. $\boxtimes$

Dimostrazione. Sia:

$$S \stackrel{def}{=} \{\ w \in D \mid \forall x, y \geq 0 \quad w\,(x, y) \neq \bot \quad \Rightarrow \quad w\,(x, y) = (0, (x!) * y)\ \}$$

S è ammissibile; quindi, per provare la proprietà (8.4), grazie al principio di induzione di Scott è sufficiente verificare che:

$$w \in S \Rightarrow \forall w \in D \ f(w) \in S$$

Si supponga che $w \in S$, $x, y \geq 0$ e $f(w)(x,y) \neq \perp$. Bisogna verificare che $f(w)(x,y) = (0, (x!) * y)$.

Consideriamo i due casi $x = 0$ e $x > 0$ separatamente.

- Se $x = 0$, per (8.3):

$$f(w)(x,y) = (x,y) = (0,y) = (0, 1 * y) = (0, (!0) * y) = (0, (x!) * y)$$

Se $x > 1$, per (8.3):

$$w(x-1, x*y) = f(w)(x,y) \neq \perp \text{ (per assunzione)}$$

Poiché $w \in S$ e $x - 1, x * y \geq 0$:

$$w(x-1, x*y) = (0, (x-1)! * (x*y))$$

Quindi:

$$f(w)(x,y) = w(x-1, x*y) = (0, (x-1)! * (x*y)) = (0, (x!) * y)$$

□

La difficoltà nell'applicazione del principio di induzione di Scott sui punti fissi, in alcuni particolari casi, consiste nell'identificazione di un appropriato sottoinsieme ammissibile S i.e. la ricerca di "un'ipotesi di induttiva" adeguatamente forte. Il prossimo esempio illustra uno di questi casi particolari.

Esempio 8.0.30. Sia D un dominio e siano p, h e k tre funzioni continue definite come:

- $p : D \to \mathbb{B}_\perp$
- $h, k : D \to D$

dove h è esatta (i.e. $h(\perp) = \perp$).

Inoltre, siano $f_1, f_2 : (D \times D) \to D$ funzioni continue tali che $\forall d_1, d_2 \in D$:

$$f_1(d_1, d_2) = if(p(d_1), d_2, h(f_1(k(d_1), d_2)))$$
$$f_2(d_1, d_2) = if(p(d_1), d_2, f_2(k(d_1), h(d_2)))$$

dove:

$$if\,(b, d_1, d_2) = \begin{cases} d_1 & se\ b = true \\ d_2 & se\ b = false \\ \perp & se\ b = \perp \end{cases}$$

Allora $f_1 = f_2$. ⊠

Dimostrazione. Per la dimostrazione è necessario definire alcune funzioni.

Sia $E : (D \times D) \to D$,

$$g \stackrel{def}{=} \langle g_1, g_2 \rangle : (E \times E) \to (E \times E)$$

dove $g_1, g_2 : (E \times E) \to E$ sono funzioni continue definite come segue:

$$g_1\,(u_1, u_2)\,(d_1, d_2) \stackrel{def}{=} \begin{cases} d_2 & \text{se } p\,(d_1) = true \\ h\,(u_1\,(k\,(d_1), d_2)) & \text{se } p\,(d_1) = false \\ \perp & \text{se } p\,(d_1) = \perp \end{cases}$$

$$g_2\,(u_1, u_2)\,(d_1, d_2) \stackrel{def}{=} \begin{cases} d_2 & \text{se } p\,(d_1) = true \\ u_2\,(k\,(d_1), h\,(d_2)) & \text{se } p\,(d_1) = false \\ \perp & \text{se } p\,(d_1) = \perp \end{cases}$$

dove $u_1, u_2 \in E$ e $d_1, d_2 \in D$.

Per definizione di f_1 e f_2 si ha $(f_1, f_2) = fix\,(g)$. Quindi, per provare $f_1 = f_2$ è sufficiente verificare che $fix\,(g)$ appartiene al sottoinsieme ammissibile $\{(u_1, u_2) \in E \times E \mid u_1 = u_2\}$.

Per poter utilizzare il principio di induzione di Scott su questo sottoinsieme ammissibile bisogna verificare che:

$$\forall\,(u_1, u_2) \in E \times E \quad (u_1, u_2) \in \Delta \Rightarrow g\,(u_1, u_2) \in \Delta$$

i.e. $\forall u \in E \quad g_1\,(u, u) = g_2\,(u, u)$.

Per definizione di g_1 e g_2 vale:

$$g_1\,(u, u)\,(d_1, d_2) = g_2\,(u, u)\,(d_1, d_2)$$

qualora

$$h\,(u\,(k\,(d_1), d_2)) = u\,(k\,(d_1), h\,(d_2))$$

Purtroppo, questa ultima condizione non può essere ritenuta soddisfatta da un arbitrario elemento $u \in E$.

Si può risolvere il problema applicando il principio di induzione di Scott ad un sottoinsieme che sia più piccolo di $\{(u_1, u_2) \in E \times E \mid u_1 = u_2\}$:

$$S \stackrel{def}{=} \{(u_1, u_2) \in E \times E \mid u_1 = u_2 \text{ e } \forall d_1, d_2 \in D \times D \;\; h\,(u_1\,(d_1, d_2)) = u_1\,(d_1, h\,(d_2))\}$$

Bisogna per prima cosa verificare che l'insieme S è ammissibile.

S è chiuso per catene poiché se $(u_{1,0}, u_{2,0}) \sqsubseteq (u_{1,1}, u_{2,1}) \sqsubseteq (u_{1,2}, u_{2,2}) \sqsubseteq \ldots$ è una catena in $E \times E$ e ogni elemento della catena è in S, allora:

$$\bigsqcup_{n\geq 0} (u_{1,n}, u_{2,n}) = \left(\bigsqcup_{i\geq 0} u_{1,i}, \bigsqcup_{j\geq 0} u_{2,j}\right)$$

è in S, infatti:

$$\bigsqcup_{n\geq 0} u_{1,n} = \bigsqcup_{n\geq 0} u_{2,n} \qquad \text{(poiché } \forall n \;\; u_{1,n} = u_{2,n})$$

Inoltre:

$$h\left(\left(\bigsqcup_{n\geq 0} u_{1,n}\right)(d_1, d_2)\right) = h\left(\bigsqcup_{n\geq 0} u_{1,n}\,(d_1, d_2)\right)$$

h è continua e i minimi comuni maggioranti di funzione vengono calcolati argomento per argomento:

$$u_{1,n}, u_{2,n} = \bigsqcup_{n\geq 0} h\,(u_{1,n}\,(d_1, d_2))$$

Ogni $u_{1,n}, u_{2,n}$ è in S ed i minimi comuni maggioranti di funzione vengono calcolati argomento per argomento:

$$= \bigsqcup_{n\geq 0} u_{1,n}\,(d_1, h\,(d_2))$$

Quindi S è chiuso per catene/ammissibile.

Inoltre, S contiene l'elemento minimo $(\bot, \bot) \in E \times E$, poiché se $(u_1, u_2) = (\bot, \bot)$ allora $u_1 = u_2$ e:

$\forall\,(d_1, d_2) \in D \times D$

$$h\,(u_1\,(d_1, d_2)) = h\,(\bot\,(d_1, d_2))$$

h è esatta per definizione e, per definizione di $\bot \in (D \times D) \to D$:

$$= h\,(\bot)$$

$$= \perp$$

Dalla definizione di $\perp \in (D \times D) \rightarrow D$:

$$= \perp (d_1, h(d_2))$$

$$= u_1 (d_1, h(d_2))$$

Per provare che $f_1 = f_2$ bisogna verificare che $(f_1, f_2) = fix(g) \in S$. Poiché S è ammissibile, per l'induzione di Scott bisogna provare che:

$\forall (u_1, u_2) \in E \times E$:

$$(u_1, u_2) \in S \Rightarrow (g_1 (u_1, u_2), g_2 (u_1, u_2)) \in S$$

Si supponga che $(u_1, u_2) \in S$ i.e. $u_1 = u_2$

$$\forall (d_1, d_2) \in D \times D \quad h(u_1 (d_1, d_2)) = u_1 (d_1, h(d_2)) \tag{8.5}$$

Dalla definizione di g_1, g_2 si ha che $u_1 = u_2$, e per la proprietà (8.5) si ha che $g_1 (u_1, u_2) = g_2 (u_1, u_2)$. Per provare che $(g_1 (u_1, u_2), g_2 (u_1, u_2)) \in S$ bisogna verificare che $h(g_1 (u_1, u_2)(d_1, d_2)) = g_1 (u_1, u_2)(d_1, h(d_2))$ per ogni $(d_1, d_2) \in D \times D$.

Inoltre:

$$h(g_1 (u_1, u_2)(d_1, d_2)) = \begin{cases} h(d_2) & \text{se } p(d_1) = true \\ h(h(u_1 (k(d_1), d_2))) & \text{se } p(d_1) = false \\ h(\perp) & \text{se } p(d_1) = \perp \end{cases}$$

e

$$g_1 (u_1, u_2)(d_1, h(d_2)) = \begin{cases} h(d_2) & \text{se } (d_1) = true \\ h(u_1 (k(d_1), h(d_2))) & \text{se } p(d_1) = false \\ \perp & \text{se } p(d_1) = \perp \end{cases}$$

Quindi, $h(h(u_1 (k(d_1), d_2))) = h(u_1 (k(d_1), h(d_2)))$ (per la proprietà (8.5)) e $h(\perp) = \perp$, da cui segue il risultato. □

Parte III

PCF

Capitolo 9

PCF: Programming Computable Function

PCF è un semplice linguaggio di programmazione che fu ampiamente utilizzato come esempio nello sviluppo della teoria della semantica denotazionale ed operazionale. La sua sintassi fu introdotta da Dana Scott nel 1969 come parte di [Sco93] e fu studiata come linguaggio di programmazione nel 1977 da Plotkin ([Plo77]). Sommariamente, si può vedere il PCF come una sorta di λ-calcolo tipato.

Ora si presentano la sintassi e la semantica operazionale di PCF. Successivamente si illustrerà la semantica denotazionale utilizzando dominî e funzioni continue.

Tabella 9.1: Gordon D. Plotkin

Tratto da [Wik06].
Gordon D. Plotkin, informatico, ha sviluppato la *semantica operazionale strutturale* (SOS) ed ha lavorato anche sulla *semantica denotazionale*. Molto influenti sono state le sue note *A Structural Approach to Operational Semantics* del 1981. È professore di Informatica Teorica all'*università di Edinburgo* in Scozia, dove è co-fondatore del *laboratorio per la fondazione di informatica* (LFCS).

9.1 Termini e tipi

Si presentano ora le definizioni sintattiche principali di PCF.

Definizione 9.1.1 (Sintassi tipi)**.** *La sintassi per i tipi è la seguente:*

$$\tau \ ::= \ nat \mid bool \mid \tau \to \tau$$

$\Diamond$

Definizione 9.1.2 (Sintassi espressioni)**.** *La sintassi per le espressioni è la seguente:*

$$\begin{aligned} M ::= \quad & \mathtt{0} \mid \mathtt{succ}(M) \mid \mathtt{pred}(M) \mid \mathtt{zero}(M) \mid \mathtt{true} \mid \mathtt{false} \mid x \mid \\ & \mathtt{if}\ M\ \mathtt{then}\ M\ \mathtt{else}\ M \mid \\ & \mathtt{fn}\ x : \tau . M \ \mid \\ & M\ M \mid \\ & \mathtt{fix}(M) \end{aligned}$$

dove $x \in V$, e V è un insieme infinito di variabili. $\Diamond$

Nel seguito sarà sempre considerata l'α-equivalenza i.e. due espressioni distinte possono essere eguagliate sintatticamente per α-conversione di variabili vincolate (create dal termine `fn`). Quindi, un termine PCF denota una classe di espressioni $\alpha - equivalenti$, i cui elementi differiscono solo per il nome delle variabili legate.

Alcune osservazioni sulla sintassi appena introdotta:

- *nat* è il tipo dei numeri naturali i.e. denota i valori 0, 1, 2, 3, ... In PCF questi sono generati partendo dal numero `0` ed iterando l'operatore successore – $\mathtt{succ}(n)$ – il quale aggiunge 1 al suo argomento. L'operatore predecessore – $\mathtt{pred}(n)$ – sottrae 1 all'argomento n ed è indefinito per `0`.
- *bool* è il tipo booleano i.e. denota i valori `true` e `false`. L'operazione $\mathtt{zero}(n)$ verifica che l'argomento sia `0` e restituisce `true` o `false`. L'espressione *condizionale* $\mathtt{if}\ M_1\ \mathtt{then}\ M_2\ \mathtt{else}\ M_3$ si comporta come M_2 oppure M_3 in funzione del valore di verità di M_1.
- $\tau \to \tau'$ è il tipo delle funzioni (parziali) che hanno un singolo argomento di tipo τ e restituiscono (se possibile) un risultato di tipo τ'. $\mathtt{fn}\ x : \tau . M$ è la notazione per l'astrazione di funzioni[1]; si osservi che il tipo τ della variabile astratta x è fornito esplicitamente. L'applicazione della funzione M_1 all'argomento M_2 è denotata con $M_1 M_2$. L'ambito di una astrazione

[1] i.e. la λ-astrazione nel λ-calcolo.

di funzione si estende quanto più possibile alla destra del punto e l'applicazione di funzioni associa a sinistra e.g. $M_1 M_2 M_3$ equivale a $(M_1\ M_2)M_3$ e non a $M_1(M_2\ M_3)$.

- Una variabile PCF x è un'espressione sconosciuta. PCF è un linguaggio funzionale puro i.e. non ci sono stati che variano durante la valutazione dell'espressione, ed in particolare le variabili sono identificatori che denotano espressioni 'statiche' piuttosto che variabili il cui contenuto potrebbe cambiare durante la valutazione.

- L'espressione $\texttt{fix}(M)$ denota un elemento x definito ricorsivamente da $x = Mx$. Il λ-termine equivalente è YM, dove Y è un opportuno combinatore di punto fisso.

9.2 Variabili libere, variabili vincolate e sostituzione

In PCF si ha un'istruzione per legare variabili: occorrenze libere di x in M diventano legate con $\texttt{fn}\ x : \tau.M$.

Definizione 9.2.1 (Variabili libere)**.** *L'insieme (finito) delle variabili libere di un'espressione M, $fv(M)$, è definito per induzione sulla sua struttura:*

$$\begin{aligned}
fv(\texttt{0}) = fv(\texttt{true}) = fv(\texttt{false}) &\stackrel{def}{=} \emptyset \\
fv(x) &\stackrel{def}{=} \{x\} \\
fv(\texttt{succ}(M)) = fv(\texttt{pred}(M)) = fv(\texttt{zero}(M)) = fv(\texttt{fix}(M)) &\stackrel{def}{=} fv(M) \\
fv(\texttt{if}\ M\ \texttt{then}\ M'\ \texttt{else}\ M'') &\stackrel{def}{=} fv(M) \cup fv(M') \cup fv(M'') \\
fv(MM') &\stackrel{def}{=} fv(M) \cup fv(M') \\
fv(\texttt{fn}\ x : \tau.M) &\stackrel{def}{=} fv(M) \setminus \{x\}
\end{aligned}$$

$\Diamond$

Definizione 9.2.2 (Termine chiuso, aperto)**.** *Un termine M è* chiuso *se $fv(M) = \emptyset$,* aperto *altrimenti.* $\Diamond$

Come detto, in PCF si considera l'α-equivalenza; questa induce una partizione dei termini. Quindi, ci si può riferire ad un termine mediante un'espressione rappresentativa, tipicamente scegliendone una le cui variabili legate sono tutte distinte tra loro e da qualsiasi altra variabile del contesto in cui il termine è posto.

Definizione 9.2.3 (Sostituzione)**.** *Si denota l'operazione di* sostituzione *di un termine M, in tutte le occorrenze libere di una variabile x in un termine M', con $M'[M/x]$.* ◊

Tale operazione consiste nella sostituzione di M in ogni occorrenza libera di x nel termine M'; le variabili legate di M' devono essere disgiunte dalle variabili libere di M (altrimenti si potrebbero legare delle variabili libere di M).

9.3 Tipi

Definizione 9.3.1 (Ambiente di tipi)**.** *Un* ambiente di tipi Γ *è una funzione parziale finita che mappa variabili in tipi.* ◊

I tipi sono assegnati ai termini mediante la seguente relazione.

Definizione 9.3.2 (Relazione di tipaggio)**.** *$M : \tau$ denota la situazione in cui M è chiuso e $\emptyset \vdash M : \tau$; inoltre, $PCF_\tau \stackrel{def}{=} \{M \mid M : \tau\}$. $\Gamma \vdash M : \tau$ è la relazione di tipaggio: $\forall x \in dom(\Gamma)\ \ x : \Gamma(x) \Rightarrow M : \tau$.* ◊

Definizione 9.3.3 (Regole di tipaggio)**.** *La relazione di tipaggio è definita induttivamente dagli assiomi e dalle regole che seguono.*

$$\frac{}{\Gamma \vdash \mathtt{0} : nat}\ (:_0) \qquad \frac{\Gamma \vdash M : nat}{\Gamma \vdash \mathtt{succ}(M) : nat}\ (:_{succ})$$

$$\frac{\Gamma \vdash M : nat}{\Gamma \vdash \mathtt{pred}(M) : nat}\ (:_{pred}) \qquad \frac{\Gamma \vdash M : nat}{\Gamma \vdash \mathtt{zero}(M) : bool}\ (:_{zero})$$

$$\frac{}{\Gamma \vdash b : bool}\ (:_{bool})\ \ (b \in \{\mathtt{true}, \mathtt{false}\}) \qquad \frac{x \in dom(\Gamma) \wedge \Gamma(x) = \tau}{\Gamma \vdash x : \tau}\ (:_{var})$$

$$\frac{\Gamma \vdash M_1 : bool \quad \Gamma \vdash M_2 : \tau \quad \Gamma \vdash M_3 : \tau}{\Gamma \vdash \mathtt{if}\ M_1\ \mathtt{then}\ M_2\ \mathtt{else}\ M_3 : \tau}\ (:_{if}) \qquad \frac{\Gamma\,[x \mapsto \tau] \vdash M : \tau'}{\Gamma \vdash \mathtt{fn}\ x : \tau . M : \tau \to \tau'}\ (:_{fn})\ \ (x \notin dom(\Gamma))$$

$$\frac{\Gamma \vdash M_1 : \tau \to \tau' \quad \Gamma \vdash M_2 : \tau}{\Gamma \vdash M_1 M_2 : \tau'}\ (:_{app}) \qquad \frac{\Gamma \vdash M : \tau \to \tau'}{\Gamma \vdash \mathtt{fix}(M) : \tau}\ (:_{fix})$$

In $(:_{fn})$, $\Gamma\,[x \mapsto \tau]$ denota l'ambiente di tipi che mappa x in τ e si comporta come Γ negli altri casi. ◊

Si osservi che non tutti i termini sono tipabili e.g. `succ(true)`. Si considerano *corretti* i termini tipabili.

Si presentano ora alcuni risultati sul tipaggio. Per quanto riguarda le dimostrazioni è stato scelto il livello di dettaglio utilizzato in aula: alcune dimostrazioni (le più difficili e delicate) saranno presentate per intero; per altre, in

particolare le dimostrazioni per induzione strutturale, saranno presentati solo i casi principali. In molte dimostrazioni, si è ritenuto tuttavia opportuno approfondire il livello di dettaglio tenuto a lezione. Le dimostrazioni complete sono comunque reperibili nei testi citati in bibliografia.

Proposizione 9.3.4. *Se* $\Gamma \vdash M : \tau$ *allora* $fv(M) \subseteq dom(\Gamma)$. $\Diamond$

Dimostrazione(cenni). Per induzione strutturale sulle regole 9.3.3.

Per i casi base (gli assiomi) è diretta.

($:_{var}$) Si ha $M = x$, quindi si deve mostrare che $\Gamma \vdash x : \tau \Rightarrow fv(x) = \{x\} \subseteq dom(\Gamma) \Leftrightarrow x \in dom(\Gamma)$; ciò è vero poiché è fra le premesse della regola.

($:_{succ}$), ($:_{pred}$), ($:_{zero}$) Per IH $fv(M) \subseteq \Gamma$ quindi $fv(\texttt{succ}(M)) = fv(\texttt{pred}\rightarrow(M)) = fv(\texttt{zero}(M)) = fv(M) \subseteq \Gamma$.

($:_{if}$) Per IH $fv(M_1) \subseteq \Gamma$, $fv(M_2) \subseteq \Gamma$, $fv(M_3) \subseteq \Gamma$; quindi $fv(M) = fv(M_1) \cup fv(M_2) \cup fv(M_3) \subseteq \Gamma$.

($:_{fn}$) Per IH $fv(M) \subseteq dom(\Gamma)$; $M' =$`fn` $x : \tau.M$, quindi $fv(M') = fv(M) \setminus \{x\} \subseteq dom(\Gamma)$.

Per gli altri casi si procede analogamente.

□

Proposizione 9.3.5. *Se* $\Gamma \vdash M : \tau$ *e* $\Gamma \vdash M : \tau'$ *allora* $\tau = \tau'$. $\Diamond$

Dimostrazione(cenni). Se M è una costante, vale banalmente. Se M è una variabile, essendo Γ una funzione, essa associa un unico valore. Per `fn` si osservi che si utilizza il fatto che il tipo è assegnato esplicitamente. □

Proposizione 9.3.6. *Se* $\Gamma \vdash M : \tau$ *e* $\Gamma\,[x \mapsto \tau] \vdash M' : \tau'$ *allora* $\Gamma \vdash M'[M/x] : \tau'$. $\Diamond$

Dimostrazione(cenni). Per induzione strutturale. Gli assiomi non hanno sostituzioni. Qualora M' sia una variabile: $M' = x, x \mapsto \tau \Rightarrow M' = \tau = \tau'$. □

Si osservi che l'assegnamento è parziale, come si evince dai seguenti esempi.

Esempio 9.3.7 (Tipaggio)**.**

- `succ(succ 0)`: *nat*
- `if zero` x `then succ` x `else pred` x : *nat* $\Leftrightarrow$ x : *nat* (i.e. dipende dal tipo di x)
- `if zero (zero` x `) then` x `else pred` x non ha tipo

⊠

Esempio 9.3.8 (Funzioni parziali ricorsive in PCF)**.** Sebbene la sintassi di PCF sia piuttosto succinta, esso gode della proprietà di essere espressivo come una Macchina di Turing, ed in particolare tutte le funzioni ricorsive parziali possono essere codificate.

Ad esempio, la funzione parziale $h : \mathbb{N} \times \mathbb{N} \rightharpoonup \mathbb{N}$ definita attraverso la *ricorsione primitiva* da $f : \mathbb{N} \rightharpoonup \mathbb{N}$ e $g : \mathbb{N} \times \mathbb{N} \times \mathbb{N} \rightharpoonup \mathbb{N}$ soddisfa $\forall x, y \in \mathbb{N}$:

$$\begin{cases} h(x,0) &= f(x) \\ h(x,y+1) &= g(x,y,h(x,y)) \end{cases}$$

Così, se la funzione f viene codificata in PCF dal termine $F : nat \to nat$ e g dal termine $G : nat \to (nat \to (nat \to nat))$[2], allora h può essere codificata nel seguente modo:

$H \stackrel{def}{=}$

```
fix(
  fn h : nat → (nat → nat) fn x : nat fn y : nat
    if zero(y) then Fx else G x pred(y)(h x pred(y))
)
```

⊠

Esempio 9.3.9 (Minimizzazione)**.** Oltre alla ricorsione primitiva l'altro costrutto necessario per definire funzioni ricorsive parziali è la *minimizzazione* e.g. la funzione parziale $m : \mathbb{N} \rightharpoonup \mathbb{N}$ definita da $k : \mathbb{N} \times \mathbb{N} \rightharpoonup \mathbb{N}$ attraverso la minimizzazione, soddisfa per ogni $x \in \mathbb{N}$:

$$\begin{aligned} m(x) = &\text{ il più piccolo } y \geq 0 \text{ tale che} \\ &k(x,y) = 0 \text{ e} \\ &\forall z\ 0 \leq z < y \Rightarrow k(x,z) > 0 \end{aligned}$$

Ciò potrebbe anche essere espresso utilizzando punti fissi, ma non così facilmente come nel caso della ricorsione primitiva. Se k fosse codificato in PCF dal termine $K : nat \to (nat \to nat)$, allora m potrebbe essere codificato come $\texttt{fn}\ x : nat.M'\ x\ \texttt{0}$ dove:

$M' \stackrel{def}{=}$

```
fix( fn m' : nat → (nat → nat).fn x : nat.fn y : nat.
  if zero(Kxy) then y else m'x succ(y)
)
```

⊠

[2]Si usa la currificazione.

9.4 Valutazione

Si fornisce ora la semantica operazionale del linguaggio PCF come una relazione definita induttivamente.

Come notazione si ha $M \Downarrow_\tau V$[3], dove τ è un tipo PCF, $M, V \in PCF_\tau$ sono termini chiusi e V è un valore (introduciamo questo concetto nella definizione seguente).

Definizione 9.4.1 (Valore)**.**

$V ::= \mathtt{0} \mid \mathtt{succ}(V) \mid \mathtt{true} \mid \mathtt{false} \mid \mathtt{fn}\ x : \tau.M$ $\lozenge$

I risultati delle valutazioni sono termini PCF di una particolare forma, chiamati *valori* oppure *forme canoniche*. I valori di tipo *bool* sono `true` e `false`. I valori di tipo *nat* sono rappresentazioni unarie di numeri naturali i.e. $\mathtt{succ}^n(0)$ (dove $n \in \mathbb{N}$) definito come:

$$\begin{cases} \mathtt{succ}^0(0) & \stackrel{def}{=} & 0 \\ \mathtt{succ}^{n+1}(0) & \stackrel{def}{=} & \mathtt{succ}(\mathtt{succ}^n(0)) \end{cases}$$

I valori di tipo funzione, essendo astrazioni di funzione $\mathtt{fn}\ x : \tau.M$, sono più "intensionali" di quelli di tipi di base, poiché M è un termine PCF non ancora valutato.

Informalmente, si può pensare la valutazione di un termine come il "valore calcolato", e tale valore può essere sia una costante sia una funzione.

Si forniscono ora gli assiomi e le regole per la valutazione.

Definizione 9.4.2 (Regole valutazione)**.**

[3]Si legge "M valuta a V".

$$\frac{}{V \Downarrow_\tau V}\ (\Downarrow_{val})\quad V \text{ valore di tipo } \tau \qquad \frac{M \Downarrow_{nat} V}{\mathtt{succ}(M) \Downarrow_{nat} \mathtt{succ}(V)}\ (\Downarrow_{succ})$$

$$\frac{M \Downarrow_{nat} \mathtt{succ}(V)}{\mathtt{pred}(M) \Downarrow_{nat} V}\ (\Downarrow_{pred}) \qquad \frac{M \Downarrow_{nat} 0}{\mathtt{zero}(M) \Downarrow_{bool} \mathtt{true}}\ (\Downarrow_{zero1})$$

$$\frac{M \Downarrow_{nat} \mathtt{succ}(V)}{\mathtt{zero}(M) \Downarrow_{bool} \mathtt{false}}\ (\Downarrow_{zero2}) \qquad \frac{M_1 \Downarrow_{bool} \mathtt{true} \quad M_2 \Downarrow_\tau V}{\mathtt{if}\ M_1\ \mathtt{then}\ M_2\ \mathtt{else}\ M_3 \Downarrow_\tau V}\ (\Downarrow_{if1})$$

$$\frac{M_1 \Downarrow_{bool} \mathtt{false} \quad M_3 \Downarrow_\tau V}{\mathtt{if}\ M_1\ \mathtt{then}\ M_2\ \mathtt{else}\ M_3 \Downarrow_\tau V}\ (\Downarrow_{if2}) \qquad \frac{M_1 \Downarrow_{\tau\to\tau'} \mathtt{fn}\, x:\tau.M_1' \quad M_1'[M_2/x] \Downarrow_{\tau'} V}{M_1 M_2 \Downarrow_{\tau'} V}\ (\Downarrow_{cbn})$$

$$\frac{M\,\mathtt{fix}(M) \Downarrow_\tau V}{\mathtt{fix}(M) \Downarrow_\tau V}\ (\Downarrow_{fix})$$

$\Diamond$

Proposizione 9.4.3 (Valutazione deterministica)**.** *La valutazione è* deterministica*: se $M \Downarrow_\tau V$ e $M \Downarrow_\tau V'$ allora $V = V'$.* $\Diamond$

Dimostrazione (cenni). Per induzione sulla complessità del termine. L'idea chiave è che ogni termine ha un'unica scrittura.

0, `succ`, `pred`, `zero`, `true`, `false` e `fn` sono valori, per $\Downarrow_{val}$ e l'unicità della scrittura si ha la tesi.

Si può pensare che si potrebbe avere duplicazione qualora si avessero questi termini come conseguenze, ma valendo l'unicità per la premessa, la verifica per la conseguenza è diretta e.g. in $(\Downarrow_{succ})$: $(M \Downarrow_{nat} V \wedge M \Downarrow_{nat} V' \Rightarrow V = V') \Rightarrow succ(M) \Downarrow_{nat} succ(V) = succ(V')$. Per gli altri casi è analogo. □

Osservazione 9.4.4 (Relazione parziale). La relazione di valutazione è parziale. $\Diamond$

Si verifica la precedente osservazione col seguente esempio.

Esempio 9.4.5 (Relazione parziale)**.** La precedente proposizione mostra che ogni termine tipabile chiuso è valutato al più ad un valore. Tuttavia, ci sono alcuni termini tipabili che non possono essere valutati.

Si dice che M *non valuta* ($M \not\Downarrow_\tau$) sse $M : \tau \ \wedge\ \nexists V\ M \Downarrow_\tau V$.

e.g. sia $\Omega_\tau \stackrel{def}{=} \mathtt{fix}(\mathtt{fn}\, x : \tau.x)$.

Risulta $\Omega_\tau \not\Downarrow_\tau$.

Infatti, si supponga - per assurdo - che per qualche V esista una dimostrazione di $\mathtt{fix}(\mathtt{fn}\, x : \tau.x) \Downarrow_\tau V$; sia $\mathcal{D}$ la dimostrazione di peso minimo.

Allora $\mathcal{D}$ avrà necessariamente una struttura analoga alla seguente (informalmente, partendo dal basso si può osservare che le uniche regole "utili" sono quelle utilizzate).

$$\dfrac{\dfrac{\dfrac{}{\texttt{fn}\ x:\tau.x \Downarrow \texttt{fn}\ x:\tau.x}\ (\Downarrow_{val}) \quad \overset{\mathcal{D}'}{\texttt{fix}(\texttt{fn}\ x:\tau.x) \Downarrow V}}{(\texttt{fn}\ x:\tau.x)(\texttt{fix}(\texttt{fn}\ x:\tau.x)) \Downarrow V}\ (\Downarrow_{cbn})}{\texttt{fix}(\texttt{fn}\ x:\tau.x) \Downarrow V}\ (\Downarrow_{fix})$$

Dove $\mathcal{D}'$ è una dimostrazione strettamente più corta di $\texttt{fix}(\texttt{fn}\, x:\tau.x) \Downarrow_\tau V$; ciò contraddice la minimalità di $\mathcal{D}$.

Inoltre, tale dimostrazione procederebbe all'infinito: si ha un regresso all'infinito i.e. applicando le regole ci si porta allo stesso problema di partenza.
⊠

La valutazione può essere definita in termini di una transizione ad un passo (quindi si evidenzia in particolare "l'operazionalità" di tale semantica). Tale relazione viene definita di seguito.

Definizione 9.4.6 (Relazione di transizione)**.**

$$\dfrac{M \to_{nat} M'}{op(M) \to_\tau op(M')} \quad con\ ((op \in \{\texttt{succ},\texttt{pred}\}\ \wedge\ \tau = nat) \vee (op = \texttt{zero} \wedge \tau = bool))$$

$\texttt{zero(0)} \to_{bool} \texttt{true}$

$\texttt{zero(succ(V))} \to_{bool} \texttt{false}$ con V di tipo nat. $\qquad \texttt{pred(succ(V))} \to_{nat} \texttt{V}$

$$\dfrac{M_1 \to_{bool} M_1'}{\texttt{if}\ M_1\ \texttt{then}\ M_2\ \wedge\ M_3 \to_\tau \texttt{if}\ M_1'\ \texttt{then}\ M_2\ \texttt{else}\ M_3} \qquad \texttt{if true then}\ M_1\ \texttt{else}\ M_2\ \to_\tau\ M_1$$

$\texttt{if false then}\ M_1\ \texttt{else}\ M_2\ \to_\tau\ M_2$ $\qquad \dfrac{M_1 \to_{\tau\to\tau'} M_1'}{M_1M_2 \to_{\tau'} M_1'M_2}$

$(\texttt{fn}\, x:\tau.M_1)M_2 \to_\tau M_1[M_2/x]$ $\qquad \texttt{fix}(M) \to_\tau M\ \texttt{fix}(M)$

◊

Si presenta ora un teorema che evidenzia il fatto che una valutazione può essere vista come una sequenza di transizioni.

Teorema 9.4.7. *Per ogni tipo τ e per ogni coppia di termini $M, V \in PCF_\tau$*

$$M \Downarrow_\tau V \Leftrightarrow M \to_\tau^* V$$

dove V è un valore[4]*.* $\Diamond$

Dimostrazione (cenni). Per induzione strutturale.

($\Rightarrow$)

($\Downarrow_{val}$) $V \to^* V$ vero per riflessività.

($\Downarrow_{succ}$) IH: $M \to^*_{nat} V \Rightarrow$ $\texttt{succ}\ M \to^*_{nat} \texttt{succ}\ V$ (è la regola vista).

($\Downarrow_{pred}$) IH: $M \to^*_{nat} \texttt{succ}\ V \Rightarrow$ $\texttt{pred}\ M \to^*_{nat} \texttt{pred(succ}\ V) = V$.

($\Downarrow_{zero1}$) IH: $M \to^*_{nat} \texttt{0} \Rightarrow$ $\texttt{zero}\ M \to \texttt{zero 0} \to \texttt{true}$.

Per gli altri casi è analogo.

($\Leftarrow$)

"Simmetrica" e.g.

($\Downarrow_{zero1}$) $\texttt{zero 0} \to_{bool} \texttt{true} \Rightarrow$

$$\frac{\texttt{0} \Downarrow_{nat} \texttt{0}}{\texttt{zero(0)} \Downarrow_{bool} \texttt{true}}$$

□

9.5 Equivalenza contestuale e uguaglianza nella denotazione

Ora si introdurrà la nozione di equivalenza. Si sa che il fornire una semantica corrisponde a dare un metodo di interpretazione:

$$Sintassi \overset{interpr}{\longrightarrow} Semantica$$

Può capitare che a termini sintatticamente diversi corrisponda lo stesso termine semantico e.g. una funzione (pensata estensionalmente come insieme di coppie) può essere definita intensionalmente (e sintatticamente) in più modi. Quindi, si cerca una definizione di equivalenza che consideri come "lo stesso oggetto" termini come quelli dell'esercizio precedente.

Fondamentalmente, con l'interpretazione si introduce una relazione di equivalenza sui termini della sintassi i.e. fornire una semantica corrisponde ad introdurre una equivalenza sulla sintassi.

Esempio 9.5.1 (Equivalenza)**.** $\texttt{fn}\ x : \tau.x$ e $\texttt{fn}\ y : \tau.y$ sono sintatticamente diversi ma sono semanticamente la stessa funzione. ⊠

[4] $\to^*_\tau$ è la chiusura riflessiva e transitiva della relazione $\to_\tau$.

Dopo aver introdotto la semantica operazionale, il nostro obiettivo è costruire la semantica denotazionale di PCF.

Gli obiettivi della semantica denotazionale sono i seguenti.

1. Tipi $\tau \mapsto$ dominî $[\![\tau]\!]$.
2. Termini chiusi $M : \tau \mapsto$ elementi $[\![M]\!] \in [\![\tau]\!]$[5]
3. Composizionalità: $[\![M]\!] = [\![M']\!] \Rightarrow [\![\mathcal{C}\,[M]]\!] = [\![\mathcal{C}\,[M']]\!]$
4. Correttezza: $\forall \tau \;\; M \Downarrow_\tau \; V \Rightarrow [\![M]\!] = [\![V]\!]$.
5. Adeguatezza: $[\![M]\!] = [\![V]\!] \in [\![\tau]\!] \Rightarrow M \Downarrow_\tau V$ dove $\tau \in \{\texttt{bool},\texttt{nat}\}$

Le proprietà di correttezza e di adeguatezza forniscono la connessione che stiamo cercando tra la semantica operazionale e denotazionale. Si osservi che la proprietà di adeguatezza coinvolge solo i tipi di dati piatti *nat* o *bool*. Non ci si può aspettare che una tale proprietà coinvolga anche tipi funzione a causa della natura intensionale dei valori di tali tipi. Infatti, l'adeguatezza dei tipi funzione contraddirebbe le proprietà di composizionalità e di correttezza che si desiderano per $[\![-]\!]$, come mostra il prossimo esempio.

Esempio 9.5.2. Consideriamo i seguenti termini di tipo $nat \to nat$:

$$\begin{array}{rcl} V & \stackrel{def}{=} & \texttt{fn}\; x : nat.(\texttt{fn}\; y : nat.y)\texttt{0} \\ V' & \stackrel{def}{=} & \texttt{fn}\; x : nat.0 \end{array}$$

Come detto, l'adeguatezza non può essere richiesta per le funzioni. Infatti, entrambi i termini associano 0, ma non sono la stessa funzione: $V \not\Downarrow_\tau V'$ poiché un valore riduce solo a se stesso (questo è dimostrabile). Però il comportamento è lo stesso, quindi si vuole avere la possibilità di sostituirle nello stesso contesto.

Si ha che $V \not\Downarrow_{nat\to nat} V'$, poiché per l'assioma $(\Downarrow_{val})$, $V \Downarrow_{nat\to nat} V \neq V'$ e per la proposizione (9.4.3) si ha che la valutazione è deterministica. Tuttavia, le proprietà di correttezza e composizionalità di $[\![-]\!]$ implicano che $[\![V]\!] = [\![V']\!]$.

Utilizzando gli assiomi $(\Downarrow_{val})$ e $(\Downarrow_{cbn})$ si ha che: $(\texttt{fn}\, y : nat.y)\;\texttt{0} \;\Downarrow_{nat}\; \texttt{0}$.

Quindi, per la proprietà di correttezza: $[\![(\texttt{fn}\, y : nat.y)\texttt{0}]\!] = [\![\texttt{0}]\!]$.

Allora, per la proprietà della composizionalità, siccome $\mathcal{C}[-] \stackrel{def}{=} \texttt{fn}\, x : nat.-\!\!\rightarrow$, si ha che: $[\![\mathcal{C}[(\texttt{fn}\, y : nat.y)\texttt{0}]]\!] = [\![\mathcal{C}[\texttt{0}]]\!]$ i.e. $[\![V]\!] = [\![V']\!]$. ⊠

Quindi, non si ha $V \not\Downarrow_\tau V'$, ma $[\![V]\!] = [\![V']\!]$ i.e. i due valori non sono la stessa cosa in sintassi ma lo sono in semantica. Si può dire che la semantica

[5]In generale, le denotazioni di termini aperti saranno funzioni continue.

identifica più cose della sintassi e.g. ci sono funzioni estensionalmente uguali ma intensionalmente diverse. Nell'esempio si hanno una funzione con due argomenti ed una con uno, diverse sia sintatticamente che semanticamente, ma con lo stesso valore. Serve quindi la definizione di contesto.

Definizione 9.5.3 (Contesto)**.** *I contesti PCF sono generati attraverso la grammatica per le espressioni aumentata dal simbolo '-', rappresentante un segnaposto che può essere riempito con un termine i.e.*

$$\mathcal{C} ::= - \mid \mathtt{0} \mid \mathtt{succ}(\mathcal{C}) \mid \mathtt{pred}(\mathcal{C}) \mid \mathtt{zero}(\mathcal{C}) \mid \mathtt{true} \mid \mathtt{false} \mid \mathtt{if}\ \mathcal{C}\ \mathtt{then}\ \mathcal{C}\ \mathtt{else}\ \mathcal{C} \mid x \mid \mathtt{fn}\ x : \tau.\mathcal{C} \mid \mathcal{C}\mathcal{C} \mid \mathtt{fix}(\mathcal{C})$$

◊

Dato un contesto $\mathcal{C}$, $\mathcal{C}[M]$[6] denota l'espressione che risulta dalla sostituzione di tutte le occorrenze di '-' in $\mathcal{C}$ con M. Questa forma di sostituzione può implicare la cattura delle variabili libere in M vincolate in $\mathcal{C}$ e.g. se $\mathcal{C}$ è $\mathtt{fn}\ x : \tau.-$ allora $\mathcal{C}[x]$ è $\mathtt{fn}\, x : \tau.x$.

Tuttavia, è possibile dimostrare che se M e M' sono $\alpha - convertibili$, lo sono anche $\mathcal{C}[M]$ e $\mathcal{C}[M']$. Quindi, l'operazione sulle espressioni che mappa M in $\mathcal{C}[M]$ induce un'operazione ben definita sui termini (classi di espressioni $\alpha - equivalenti$).

Definizione 9.5.4 (Equivalenza contestuale)**.** *Due frasi di un linguaggio di programmazione sono* contestualmente equivalenti *se qualunque occorrenza della prima frase in un* programma completo *può essere sostituita dalla seconda frase senza influire sui risultati osservabili dell'esecuzione del programma.* ◊

Questa nozione ricorda l'*equivalenza contestuale* delle frasi di un linguaggio di programmazione. Infatti, si ha in realtà una famiglia di nozioni, parametrizzata dalle particolari scelte prese riguardo a cosa è un "programma" nel linguaggio e a quali sono i "risultati osservabili" della loro esecuzione. Per PCF è ragionevole scegliere come programmi termini chiusi del tipo *nat* o *bool* e come valori osservabili i risultati della valutazione di tali termini.

Questo porta alla seguente definizione.

Definizione 9.5.5 (Equivalenza contestuale di termini PCF)**.** *Siano i termini M_1, M_2, un tipo τ e un ambiente di tipi Γ, si ha la relazione*

$$\Gamma \vdash M_1 \cong_{ctx} M_2 : \tau \text{ se e solo se:}$$

1. $\Gamma \vdash M_1 : \tau$ e $\Gamma \vdash M_2 : \tau$

[6] Tipicamente si scrive $\mathcal{C}[M]$ invece di $\mathcal{C}$ per indicare il simbolo "segnaposto".

2. *per ogni contesto* $\mathcal{C}$ *tale che* $\mathcal{C}[M_1]$ *e* $\mathcal{C}[M_2]$ *sono termini chiusi di tipo* γ *(dove* $\gamma \in \{nat, bool\}$*) e per tutti i valori* $V : \gamma$

$$\mathcal{C}[M_1] \Downarrow_\gamma V \Leftrightarrow \mathcal{C}[M_2] \Downarrow_\gamma V$$

$\Diamond$

Si osservi che se M_1 ed M_2 sono dello stesso tipo allora $\mathcal{C}[M_1]$ e $\mathcal{C}[M_2]$ sono dello stesso tipo; inoltre se essi sono chiusi allora sono costanti. Nell'esempio precedente le funzioni possono essere considerate dello stesso tipo se, sostituendole in ogni contesto, il contesto ha valore costante e questo deve essere uguale (quindi equivalente).

Osservazione 9.5.6 (Notazione). Per i termini chiusi si può scrivere: $M_1 \cong_{ctx} M_2 : \tau$ per $\emptyset \vdash M_1 \cong_{ctx} M_2 : \tau$. $\Diamond$

Sebbene $\cong_{ctx}$ sia una nozione "naturale" dell'equivalenza semantica per i termini, data la semantica operazionale di PCF, è difficile lavorarci a causa della quantificazione universale su contesti che occorre nella definizione.

Come si vedrà con il prossimo teorema, se si considera una semantica denotazionale che soddisfa le proprietà di chiusura, composizionalità, correttezza e adeguatezza, la si può usare per stabilire istanze di equivalenza contestuale mostrando che i termini hanno denotazioni identiche.

In alcuni casi ciò è più facile rispetto a provare l'equivalenza contestuale direttamente dalla definizione.

Il prossimo teorema generalizza quanto detto ai termini aperti: se le funzioni continue che sono denotazioni di due termini aperti (dello stesso tipo) sono uguali, allora i termini sono contestualmente equivalenti.

Questa equivalenza considera i termini aperti ma è utilizzabile anche per i termini chiusi.

Teorema 9.5.7. *Per ogni tipo* τ *e termini chiusi* $M_1, M_2 \in PCF_\tau$*, se* $[\![M_1]\!]$ *e* $[\![M_2]\!]$ *sono elementi uguali del dominio* $[\![\tau]\!]$ *allora* $M_1 \cong_{ctx} M_2 : \tau$. $\Diamond$

Dimostrazione.

$\mathcal{C}[M_1] \Downarrow_{nat} V \Rightarrow [\![\mathcal{C}[M_1]]\!] = [\![V]\!]$ (correttezza)

$\Rightarrow [\![\mathcal{C}[M_1]]\!] = [\![V]\!]$ (composizionalità su $[\![M_1]\!] = [\![M_2]\!]$)

$\Rightarrow [\![\mathcal{C}[M_1]]\!] = [\![\mathcal{C}[M_2]]\!] = \mathcal{C}[M_2] \Downarrow_{nat} V$ (adeguatezza)

e simmetricamente. □

Si fornirà PCF una semantica denotazionale che gode della proprietà composizionale, della correttezza e dell'adeguatezza.

Capitolo 10

Semantica denotazionale

Sono state presentate nella sezione 9.5 le proprietà che deve soddisfare la semantica denotazionale: quindi ora si procederà euristicamente. Dapprima verrà definita la semantica denotazionale per tipi e termini, poi ne verranno studiate le proprietà.

10.1 Denotazione di tipi

Per ogni tipo τ si definisce un dominio $[\![\tau]\!]$ per induzione sulla sua struttura.

Definizione 10.1.1 (Semantica denotazionale di tipi)**.**

$$\begin{array}{rcl} [\![nat]\!] & \stackrel{def}{=} & \mathbb{N}_\perp \\ [\![bool]\!] & \stackrel{def}{=} & \mathbb{B}_\perp \\ [\![\tau \to \tau']\!] & \stackrel{def}{=} & [\![\tau]\!] \to [\![\tau']\!] \end{array}$$

dove $\mathbb{N} = \{0, 1, 2, 3, \ldots\}$ *e* $\mathbb{B} = \{true, false\}$. $\Diamond$

10.2 Denotazione di termini

Si analizza ora l'interpretazione dei termini. Questa deve essere un elemento del tipo del termine, ma in generale questi non ne hanno uno fissato e.g. `if`$\to$ M_1 `then` M_2 `else` M_3 dipende dai tipi di M_1, M_2, M_3 e dal valore di M_1. Quindi, il tipo viene assegnato una volta fissato Γ.

Per ogni termine M e ambiente Γ, utilizzando la proposizione (9.3.5) esiste al più un tipo τ tale che la relazione $\Gamma \vdash M : \tau$ è derivabile dagli assiomi e regole (9.3.3). Si fornisce ora una semantica denotazionale per questi termini tipabili.

Specificamente, per M e Γ come sopra, si definisce una funzione continua tra i dominî:

$$[\![\Gamma \vdash M]\!] : [\![\Gamma]\!] \to [\![\tau]\!] \tag{10.1}$$

dove τ è il tipo tale che $\Gamma \vdash M : \tau$, dove $[\![\Gamma]\!]$ è il seguente prodotto dipendente (def. 6.3.5) tra dominî:

$$[\![\Gamma]\!] \stackrel{def}{=} \prod_{x \in dom(\Gamma)} [\![\Gamma(x)]\!] \tag{10.2}$$

Per quanto riguarda l'equazione 10.2, come detto, un tipo è interpretato come un dominio, quindi l'interpretazione di Γ è il prodotto di questi dominî. Infatti, Γ indica quale dominio assegnare alle variabili (e.g. $\Gamma(x), \Gamma(y), \ldots$), e $[\![\Gamma]\!]$ indica il prodotto cartesiano fra questi.

Gli elementi del dominio (10.2) saranno chiamati Γ-*ambienti*: sono funzioni ρ che mappano ogni variabile x nel dominio di definizione di Γ verso un elemento $\rho(x) \in [\![\Gamma(x)]\!]$ nel dominio che è la denotazione del tipo $\Gamma(x)$ assegnato ad x attraverso l'ambiente Γ. La funzione continua (10.1) è definita per induzione sulla struttura di M, o equivalentemente per induzione sulla derivazione del tipo $\Gamma \vdash M : \tau$. La definizione verrà presentata ora, mostrando l'effetto di ogni funzione su un Γ-ambiente, ρ.

10.2.1 Semantica denotazionale di termini PCF

Definizione 10.2.1 (Semantica denotazionale di termini PCF)**.**

$$
\begin{array}{rcl}
[\![\Gamma \vdash \mathtt{0}]\!](\rho) & \stackrel{def}{=} & 0 \in [\![nat]\!] \\
[\![\Gamma \vdash \mathtt{true}]\!](\rho) & \stackrel{def}{=} & true \in [\![bool]\!] \\
[\![\Gamma \vdash \mathtt{false}]\!](\rho) & \stackrel{def}{=} & false \in [\![bool]\!] \\
[\![\Gamma \vdash x]\!](\rho) & \stackrel{def}{=} & \rho(x) \in [\![\Gamma(x)]\!] \qquad (x \in dom(\Gamma))
\end{array}
$$

$$
[\![\Gamma \vdash \mathtt{succ}(M)]\!](\rho) \stackrel{def}{=} \begin{cases} [\![\Gamma \vdash M]\!](\rho) + 1 & se\ [\![\Gamma \vdash M]\!](\rho) \neq \bot \\ \bot & se\ [\![\Gamma \vdash M]\!](\rho) = \bot \end{cases}
$$

$$
[\![\Gamma \vdash \mathtt{pred}(M)]\!](\rho) \stackrel{def}{=} \begin{cases} [\![\Gamma \vdash M]\!](\rho) - 1 & se\ [\![\Gamma \vdash M]\!](\rho) > 0 \\ \bot & se\ [\![\Gamma \vdash M]\!](\rho) \in \{0, \bot\} \end{cases}
$$

$$
[\![\Gamma \vdash \mathtt{zero}(M)]\!](\rho) \stackrel{def}{=} \begin{cases} true & se\ [\![\Gamma \vdash M]\!](\rho) = 0 \\ false & se\ [\![\Gamma \vdash M]\!](\rho) > 0 \\ \bot & se\ [\![\Gamma \vdash M]\!](\rho) = \bot \end{cases}
$$

$$
[\![\Gamma \vdash \mathtt{if}\ M_1\ \mathtt{then}\ M_2\ \mathtt{else}\ M_3]\!](\rho) \stackrel{def}{=} \begin{cases} [\![\Gamma \vdash M_2]\!](\rho) & se\ [\![\Gamma \vdash M_1]\!](\rho) = true \\ [\![\Gamma \vdash M_3]\!](\rho) & se\ [\![\Gamma \vdash M_1]\!](\rho) = false \\ \bot & se\ [\![\Gamma \vdash M_1]\!](\rho) = \bot \end{cases}
$$

$$
\begin{array}{rcll}
[\![\Gamma \vdash M_1 M_2]\!](\rho) & \stackrel{def}{=} & ([\![\Gamma \vdash M_1]\!](\rho))([\![\Gamma \vdash M_2]\!](\rho)) & \\
[\![\Gamma \vdash \mathtt{fn}\ x : \tau . M]\!](\rho) & \stackrel{def}{=} & \lambda d \in [\![\tau]\!].[\![\Gamma[x \mapsto \tau] \vdash M]\!](\rho[x \mapsto d]) & dove\ x \notin dom(\Gamma) \\
[\![\Gamma \vdash \mathtt{fix}(M)]\!](\rho) & \stackrel{def}{=} & fix([\![\Gamma \vdash M]\!](\rho)) &
\end{array}
$$

$\Diamond$

Definizione 10.2.2 (Denotazione di termini chiusi)**.** *Sia $M \in PCF_\tau$, per definizione $\emptyset \vdash M : \tau$, quindi $[\![\emptyset \vdash M]\!] : [\![\emptyset]\!] \to [\![\tau]\!]$.*

Quando $\Gamma = \emptyset$, il solo Γ-ambiente è la funzione parziale totalmente indefinita, $\bot$.

In questo caso $[\![\Gamma]\!]$ è un dominio con un unico elemento i.e. $\{\bot\}$. Le funzioni continue $f : \{\bot\} \to D$ sono in biiezione con gli elementi $f(\bot) \in D$, ed in particolare si può identificare la denotazione dei termini chiusi con gli elementi del dominio denotando il loro tipo:

$$
[\![M]\!] \stackrel{def}{=} [\![\emptyset \vdash M]\!](\bot) \in [\![\tau]\!] \qquad (M \in PCF_\tau)
$$

$\Diamond$

10.2.2 La composizione preserva la continuità

$[\![\Gamma \vdash M]\!] : [\![\Gamma]\!] \to [\![\tau]\!]$ è una funzione continua ben definita poiché i casi base della definizione (10.2.1) sono funzioni continue e ad ogni passo induttivo, fornendo la denotazione di una frase composta in termini di denotazioni delle sue sottofrasi immediate, si possono utilizzare costruzioni che preservano la continuità.

Proposizione 10.2.3 (Continuità composizione)**.** *Siano $f : D \to E$ e $g : E \to F$ funzioni continue tra CPO, allora la loro composizione*

$$\begin{array}{rcl} g \circ f & : & D \to F \\ (g \circ f)(d) & \stackrel{def}{=} & g(f(d)) \end{array}$$

è continua. ◊

Dimostrazione.

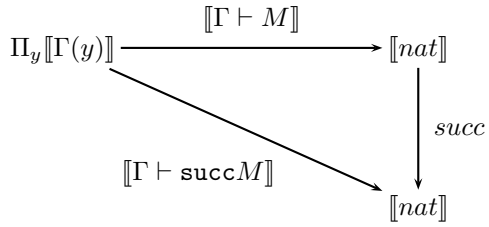
(0, `true`, `false`) Le denotazioni di questi termini sono funzioni che sono costantemente uguali ad un particolare valore e.g. dati i CPO D ed E, per ogni $e \in E$ si dimostra che la funzione costante $D \to E$ con valore e ($\lambda d \in D.e$) è continua. Praticamente si ha $\Pi_y [\![\Gamma(y)]\!] \stackrel{\Pi_x}{\to} [\![\Gamma(x)]\!]$ i.e. l'x-esima proiezione che è dimostrabile continua.

(Variabili) La denotazione di una variabile è una proiezione (def. 10.2.1). Si è visto (def. 6.3.5) che tali funzioni sono continue, poiché i minimi confini superiori sono calcolati componente per componente nel prodotto dipendente.

(`succ, pred, zero`) La denotazione $\texttt{succ}(M)$ (def. 10.2.1) è la composizione di:

$$s_\perp \circ [\![\Gamma \vdash M]\!]$$

dove per ipotesi induttiva $[\![\Gamma \vdash M]\!] : [\![\Gamma]\!] \to \mathbb{N}_\perp$ è una funzione continua, e dove $s_\perp : \mathbb{N}_\perp \to \mathbb{N}_\perp$ è la funzione continua sul dominio piatto $\mathbb{N}_\perp$, indotta (come in 6.2.4) dalla funzione $s : \mathbb{N} \rightharpoonup \mathbb{N}$ che mappa ogni n in $n + 1$.



Se $\Gamma \vdash M$ è funzione nei *nat* anche componendo con questa si ottiene *nat*. È necessario che *succ* sia continua; ciò è vero i.e. $\mathbb{N}_\perp \stackrel{succ}{\to} \mathbb{N}_\perp$, dove $\perp \mapsto \perp$ e gli altri elementi si mappano scalandoli di 1. Infine, la composizione di funzioni continue è continua.

Analogamente $[\![\Gamma \vdash \texttt{pred}(M)]\!] = p_\perp \circ [\![\Gamma \vdash M]\!]$ e $[\![\Gamma \vdash \texttt{zero}(M)]\!] = z_\perp \circ [\![\Gamma \vdash M]\!]$ per opportune funzioni $p : \mathbb{N} \nrightarrow \mathbb{N}$ e $z : \mathbb{N} \nrightarrow \mathbb{B}$.

Per `zero`:

$$\begin{array}{ccc} [\![\Pi_x \Gamma(x)]\!] & \xrightarrow{[\![\Gamma \vdash M]\!]} & [\![nat]\!] \\ & & \downarrow z \\ & & [\![bool]\!] \end{array}$$

dove $\mathbb{N}_\perp \xrightarrow{z} \mathbb{B}_\perp$ è tale che $0 \mapsto true$, $1 \mapsto false$, $2 \mapsto false$, ...

(`if-then-else`) Per IH si hanno funzioni continue $[\![\Gamma \vdash M_1]\!] : [\![\Gamma]\!] \to \mathbb{B}_\perp, [\![\Gamma \vdash M_2]\!] : [\![\Gamma]\!] \to [\![\tau]\!], [\![\Gamma \vdash M_3]\!] : [\![\Gamma]\!] \to [\![\tau]\!]$. Quindi $[\![\Gamma \vdash \texttt{if}\ M_1\ \texttt{then}\ M_2\ \texttt{else}\ M_3]\!]$ è continua poiché si può esprimere la definizione vista in precedenza in termini di composizione, proprietà di accoppiamento (def. 6.3.2) e funzione continua $\mathbb{B}_\perp \times ([\![\tau]\!] \times [\![\tau]\!]) \to [\![\tau]\!]$ (def. 6.2.5):

$$[\![\Gamma \vdash \texttt{if}\ M_1\ \texttt{then}\ M_2\ \texttt{else}\ M_3]\!] = if \circ \langle [\![\Gamma \vdash M_1]\!], \langle [\![\Gamma \vdash M_2]\!], [\![\Gamma \vdash M_3]\!] \rangle \rangle \,.$$

$$\begin{array}{ccc} [\![\Pi_x \Gamma(x)]\!] & \xrightarrow{[\![\Gamma \vdash M_1]\!]\ [\![\Gamma \vdash M_2]\!]\ [\![\Gamma \vdash M_3]\!]} & [\![bool]\!] \times [\![\tau]\!] \times [\![\tau']\!] \\ & & \downarrow if \end{array}$$

(*Applicazione*) Per IH si hanno le funzioni continue $[\![\Gamma \vdash M_1]\!] : [\![\Gamma]\!] \to ([\![\tau]\!] \to [\![\tau']\!])$ e $[\![\Gamma \vdash M_2]\!] : [\![\Gamma]\!] \to [\![\tau]\!]$. Quindi $[\![\Gamma \vdash M_1 M_2]\!]$ è continua poiché si può esprimere la definizione vista in termini di composizione, proprietà di accoppiamento e funzione $ev : ([\![\tau]\!] \to [\![\tau']\!]) \times [\![\tau]\!] \to [\![\tau']\!]$ (che è continua, cfr. (6.5.1)):

$$[\![\Gamma \vdash M_1 M_2]\!] = ev \circ \langle [\![\Gamma \vdash M_1]\!], [\![\Gamma \vdash M_2]\!] \rangle$$

IH: $[\![\Gamma M_1]\!] : \Pi [\![\Gamma(x)]\!] \to [[\![\tau]\!] \to [\![\tau]\!]]$

$[\![\Gamma M_2]\!] : \Pi [\![\Gamma(x)]\!] \to [\![\tau]\!]$

$$\begin{array}{ccc} [\![\Pi\Gamma(x)]\!] & \xrightarrow{\langle [\![\Gamma \vdash M_1]\!], [\![\Gamma \vdash M_2]\!] \rangle} & [\![\tau \to \tau']\!] \times [\![\tau]\!] \\ & & \downarrow ev \\ & & [\![\tau']\!] \end{array}$$

(*Astrazione di funzione*) Per IH si ha la funzione continua $[\![\Gamma[x \mapsto \tau] \vdash M]\!] : [\![\Gamma[x \mapsto \tau]]\!] \to [\![\tau']\!]$ con $x \notin dom(\Gamma)$. Si osservi che per ogni $\Gamma[x \mapsto \tau]$-ambiente, $\rho' \in [\![\Gamma[x \mapsto \tau]]\!]$, può essere espresso unicamente come $\rho[x \mapsto d]$, dove ρ è la restrizione della funzione ρ' a $dom(\Gamma)$ e dove $d = \rho'(x)$; inoltre l'ordine parziale rispetta questa decomposizione: $\rho_1[x \mapsto d_1] \sqsubseteq \rho_2[x \mapsto d_2]$ in $[\![\Gamma[x \mapsto \tau]]\!]$ se e solo se $\rho_1 \sqsubseteq \rho_2$ in $[\![\Gamma]\!]$ e $d_1 \sqsubseteq d_2$ in $[\![\tau]\!]$. Perciò si può identificare $[\![\Gamma[x \mapsto \tau]]\!]$ con il prodotto di dominî $[\![\Gamma]\!] \times [\![\tau]\!]$. Si può applicare l'operazione di currificazione (def. 6.5.2) per ottenere una funzione continua :

$$cur([\![\Gamma[x \mapsto \tau] \vdash M]\!]) : [\![\Gamma]\!] \to ([\![\tau]\!] \to [\![\tau']\!]) = [\![\tau \to \tau']\!]$$

che è precisamente la funzione utilizzata per definire $[\![\Gamma \vdash \texttt{fn}\ x : \tau.M]\!]$.

$$[\![\Pi\Gamma(x)]\!] \times [\![\tau]\!] \xrightarrow{\quad [\![\Gamma \vdash M]\!] \quad} [\![\tau']\!] \qquad \star \qquad [\![\tau \to \tau']\!]$$

$\star$: è in corrispondenza biunivoca (currificazione): $[\![\Gamma(x)]\!] \overset{cur(M)}{\to} [\![\tau \to \tau']\!]$.

(*Punti fissi*) Per IH si ha la funzione continua $[\![\Gamma \vdash M]\!] : [\![\Gamma]\!] \to [\![\tau \to \tau]\!]$. Ora, $[\![\tau \to \tau]\!]$ è il dominio funzione $[\![\tau]\!] \to [\![\tau]\!]$ e per definizione si ha che $[\![\Gamma \vdash \texttt{fix}(M)]\!] = fix \circ [\![\Gamma \vdash M]\!]$ è la composizione con la funzione $fix : ([\![\tau]\!] \to [\![\tau]\!]) \to [\![\tau]\!]$ che assegna i minimi punti fissi, di cui si è già dimostrata la continuità (prop. 6.6.5).

$$\begin{array}{ccc} [\![\Pi\Gamma(x_i)]\!] & \xrightarrow{\quad [\![\Gamma \vdash M]\!] \quad} & [\![\tau \to \tau']\!] \\ & & \Big\downarrow fix \\ & & [\![\tau]\!] \end{array}$$

□

10.3 Composizionalità

Il fatto che la semantica denotazionale sia *composizionale* (i.e. la denotazione di un termine è funzione della denotazione dei suoi immediati sottotermini) è una parte della definizione di $[\![\Gamma \vdash M]\!]$ per induzione sulla struttura di M. In particolare, ognuno dei modi di costruire dei termini in PCF rispetta l'uguaglianza delle denotazioni (propr. 10.3.3).

Allora la proprietà dei termini chiusi (sez. 9.5) i.e.

$$[\![M]\!] = [\![M']\!] \Rightarrow [\![\mathcal{C}[M]]\!] = [\![\mathcal{C}[M']]\!]$$

segue da questa per induzione sulla struttura del contesto $\mathcal{C}[-]$.

In generale, per termini aperti si ha la seguente proposizione.

Proposizione 10.3.1. *Sia*

$$[\![\Gamma \vdash M]\!] = [\![\Gamma \vdash M']\!] : [\![\Gamma]\!] \to [\![\tau]\!]$$

e $\mathcal{C}[-]$ *un contesto tale che* $\Gamma' \vdash \mathcal{C}[M] : \tau'$ *e* $\Gamma' \vdash \mathcal{C}[M'] : \tau'$ *per qualche tipo* τ' *e per qualche ambiente* Γ'.

Risulta:

$$[\![\Gamma' \vdash \mathcal{C}[M]]\!] = [\![\Gamma' \vdash \mathcal{C}[M']]\!] : [\![\Gamma']\!] \to [\![\tau']\!]$$

$\Diamond$

La proprietà di sostituzione di $[\![-]\!]$ fornisce un ulteriore aspetto della natura composizionale della semantica denotazionale. Ciò può essere provato per induzione sulla struttura del termine M'.

Proposizione 10.3.2 (Proprietà della sostituzione di $[\![-]\!]$)**.** *Siano*

$$\begin{array}{rcl} \Gamma & \vdash & M : \tau \\ \Gamma[x \mapsto \tau] & \vdash & M' : \tau' \end{array}$$

(per la proposizione 9.3.5) $\Gamma \vdash M'[M/x] : \tau'$

Allora, per ogni $\rho \in [\![\Gamma]\!]$*:*

$$[\![\Gamma \vdash M'[M/x]]\!](\rho) = [\![\Gamma[x \mapsto \tau] \vdash M']\!](\rho[x \mapsto [\![\Gamma \vdash M]\!]])$$

In particolare, quando $\Gamma = \emptyset$, $\quad [\![x \mapsto \tau \vdash M']\!] : [\![\tau]\!] \to [\![\tau']\!]$ *e:*

$$[\![M'[M/x]]\!] = [\![x \mapsto \tau \vdash M']\!]([\![M]\!]).$$

$\Diamond$

Si presentano ora le proprietà di composizionalità di $[\![-]\!]$:

Proprietà 10.3.3 (Composizionalità)**.**

- *se* $[\![\Gamma \vdash M]\!] = [\![\Gamma \vdash M']\!] : [\![\Gamma]\!] \to [\![nat]\!]$, *allora*
  $[\![\Gamma \vdash op(M)]\!] = [\![\Gamma \vdash op(M')]\!] : [\![\Gamma]\!] \to [\![\tau]\!]$

(dove $op \in \{\texttt{succ}, \texttt{pred}\}$ *e* $\tau = nat$ *oppure* $op = \texttt{zero}$ *e* $\tau = bool$*)*

- *se* $[\![\Gamma \vdash M_1]\!] = [\![\Gamma \vdash M_1']\!] : [\![\Gamma]\!] \to [\![bool]\!]$, $[\![\Gamma \vdash M_2]\!] = [\![\Gamma \vdash M_2']\!] : [\![\Gamma]\!] \to [\![\tau]\!]$ *e* $[\![\Gamma \vdash M_3]\!] = [\![\Gamma \vdash M_3']\!] : [\![\Gamma]\!] \to [\![\tau]\!]$ *allora*
  $[\![\Gamma \vdash \texttt{if } M_1 \texttt{ then } M_2 \texttt{ else } M_3]\!] = [\![\Gamma \vdash \texttt{if } M_1' \texttt{ then } M_2' \texttt{ else } M_3']\!] : [\![\Gamma]\!]$

- *se* $[\![\Gamma \vdash M_1]\!] = [\![\Gamma \vdash M_1']\!] : [\![\Gamma]\!] \to [\![\tau \to \tau']\!]$ *e* $[\![\Gamma \vdash M_2]\!] = [\![\Gamma \vdash M_2']\!] : [\![\Gamma]\!] \to [\![\tau]\!]$ *allora*
  $[\![\Gamma \vdash M_1 M_2]\!] = [\![\Gamma \vdash M_1' M_2']\!] : [\![\Gamma]\!] \to [\![\tau']\!]$
- *se* $[\![\Gamma[x \mapsto \tau] \vdash M]\!] = [\![\Gamma[x \mapsto \tau] \vdash M']\!] : [\![\Gamma[x \mapsto \tau]]\!] \to [\![\tau']\!]$ *allora*
  $[\![\Gamma \vdash \texttt{fn}\ x : \tau.M]\!] = [\![\Gamma \vdash \texttt{fn}\ x : \tau.M']\!] : [\![\Gamma]\!] \to [\![\tau \to \tau']\!]$
- *se* $[\![\Gamma \vdash M]\!] = [\![\Gamma \vdash M']\!] : [\![\Gamma]\!] \to [\![\tau \to \tau']\!]$ *allora*
  $[\![\Gamma \vdash \texttt{fix}(M)]\!] = [\![\Gamma \vdash \texttt{fix}(M')]\!] : [\![\Gamma]\!] \to [\![\tau]\!]$

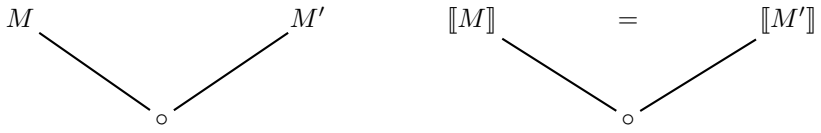
$\Diamond$

10.4 Correttezza

Un altro obiettivo indicato nella sezione 9.5 è mostrare che se un termine chiuso M valuta ad un valore V nella semantica operazionale, allora M e V hanno la stessa denotazione i.e.

$$M \Downarrow_\tau V \Rightarrow [\![M]\!] = [\![V]\!] \in [\![\tau]\!]$$

Alternativamente:



Fondamentalmente, la correttezza dice che sintassi equivalente implica semantica equivalente.

Teorema 10.4.1. *Per ogni tipo τ e per tutti i termini chiusi $M, V \in PCF_\tau$ (dove V è un valore), se $M \Downarrow_\tau V$ è derivabile dagli assiomi e regole (9.4.2) allora $[\![M]\!]$ e $[\![V]\!]$ sono elementi uguali del dominio $[\![\tau]\!]$.* $\Diamond$

Dimostrazione (cenni). Per induzione sulla definizione di $\Downarrow$. Specificamente, definendo:

$$\Phi(M, \tau, V) \overset{def}{\Leftrightarrow} [\![M]\!] = [\![V]\!] \in [\![\tau]\!]$$

si può mostrare che $\Phi(M, \tau, V)$ è chiuso rispetto agli assiomi e le regole (9.4.2). Vediamo solo i casi $\Downarrow_{cbn}$ e $\Downarrow_{fix}$

Caso ($\Downarrow_{cbn}$):

Si supponga

$$[\![M_1]\!] = [\![\texttt{fn}\ x : \tau.M_1']\!] \in [\![\tau \to \tau']\!] \tag{10.3}$$

$$[\![M_1'[M_2/x]]\!] = [\![V]\!] \in [\![\tau']\!] \tag{10.4}$$

Si deve provare che $[\![M_1 M_2]\!] = [\![V]\!] \in [\![\tau']\!]$:

$$\begin{aligned}
[\![M_1 M_2]\!] &= [\![M_1]\!]([\![M_2]\!]) && (10.2.1)\\
&= [\![\texttt{fn}\ x : \tau.M_1']\!]([\![M_2']\!]) && (10.3)\\
&= (\lambda d \in [\![\tau]\!].[\![x \mapsto \tau \vdash M_1']\!](d))([\![M_2']\!]) && (10.2.1)\\
&= [\![x \mapsto \tau \vdash M_1']\!]([\![M_2']\!])\\
&= [\![M_1'[M_2/x]]\!] && (10.3.2)\\
&= [\![V]\!] && (10.4)
\end{aligned}$$

Caso ($\Downarrow_{fix}$):

Si supponga

$$[\![M\ \texttt{fix}\ (M)]\!] = [\![V]\!] \in [\![\tau]\!] \tag{10.5}$$

Si deve provare che $[\![\texttt{fix}(M)]\!] = [\![V]\!] \in [\![\tau]\!]$:

$$\begin{aligned}
[\![\texttt{fix}(M)]\!] &= fix([\![M]\!]) && (10.2.1)\\
&= [\![M]\!]([\![\texttt{fix}([\![M]\!])) && (\text{proprietà di punto fisso di } fix)\\
&= [\![M]\!][\![\texttt{fix}(M)]\!] && (10.2.1)\\
&= [\![M\ \texttt{fix}(M)]\!] && (10.2.1)\\
&= [\![V]\!] && (10.5)
\end{aligned}$$

□

Si sono dimostrate due delle proprietà della sezione 9.5 della semantica denotazionale (necessarie per usare l'uguaglianza denotazionale per provare le equivalenze contestuali). La proprietà dell'*adeguatezza* è più difficile da provare rispetto alle prime due. La dimostrazione sarà fornita in seguito, dopo aver

introdotto una forma di principio di induzione che ci permetterà di lavorare con i minimi punti fissi.

Capitolo 11

Adeguatezza e approssimazione formale

Ora si proverà a definire la proprietà dell'*adeguatezza*. Per ogni termine chiuso M e V valore di tipo $\tau \in \{nat, bool\}$

$$[\![M]\!] = [\![V]\!] \Rightarrow M \Downarrow_\tau V$$

È un'altra forma di $[\![M_1]\!] = [\![M_2]\!] \Rightarrow M_1 \Downarrow V$ e $M_2 \Downarrow V$.

Ciò non è facile da provare. Si impiegherà un metodo creato da Plotkin nel 1977 ([Plo77]) e ripreso da Mulmuley nel 1987 ([Mul87]) in cui si fa uso della nozione di "approssimazione formale" di relazioni.

11.1 Approssimazione formale di relazioni

Si definisce una famiglia di relazioni binarie:

$$\lhd_\tau \subseteq [\![\tau]\!] \times PCF_\tau$$

indicizzate dai tipi τ. In questo modo $\lhd_\tau$ mette in relazione elementi del dominio $[\![\tau]\!]$ con termini chiusi di tipo τ. Si utilizza la notazione infissa e si scrive $d \lhd_\tau M$ al posto di $(d, M) \in \lhd_\tau$. La definizione delle relazioni $\lhd_\tau$ si ricava attraverso l'induzione sulla struttura di tipo τ e verrà fornita in seguito.

La caratteristica principale delle relazioni $\lhd_\tau$ è quella di essere rispettate dalle varie operazioni sintattiche del linguaggio PCF[1]; anche questo aspetto verrà trattato in seguito.

[1] i.e. l'applicazione delle operazioni sintattiche mantiene le proprietà di $\lhd_\tau$.

Definizione 11.1.1 ($\rho \lhd_\Gamma \sigma$)**.** *Sia* Γ *un ambiente, una* Γ- sostituzione σ *è una funzione che mappa ogni variabile* $x \in dom(\Gamma)$ *in un termine chiuso* $\sigma(x)$ *di tipo* $\Gamma(x)$*. Si ricorda che un* Γ- *ambiente* ρ *è una funzione che mappa ogni variabile* $x \in dom(\Gamma)$ *in un elemento* $\rho(x)$ *del dominio* $[\![\Gamma(x)]\!]$*. Si definisce allora:*

$$\rho \lhd_\Gamma \sigma \overset{def}{\Leftrightarrow} \forall x \in dom(\Gamma).\rho(x) \lhd_{\Gamma(x)} \sigma(x)$$

$\Diamond$

Definizione 11.1.2 ($d \lhd_\tau M$)**.** *Siano* $d \in [\![\tau]\!]$, $M \in PCF_\tau$

$$\begin{array}{rcl} d \lhd_{nat} M & \overset{def}{\Leftrightarrow} & \left(d \in \mathrm{N} \Rightarrow M \Downarrow_{nat} \mathtt{succ}^d(0)\right) \\ d \lhd_{bool} M & \overset{def}{\Leftrightarrow} & (d = true \Rightarrow M \Downarrow_{bool} \mathtt{true}) \wedge (d = false \Rightarrow M \Downarrow_{bool} \mathtt{false}) \\ d \lhd_{\tau\to\tau'} M & \overset{def}{\Leftrightarrow} & \forall e, N \ (e \lhd_\tau N \Rightarrow d(e) \lhd_{\tau'} MN) \end{array}$$

$\Diamond$

Questa è la definizione di approssimazione: come un elemento dell'interpretazione può approssimare un termine.

11.2 Proprietà fondamentali delle relazioni

Proposizione 11.2.1 (Proprietà fondamentale di $\lhd$)**.** *Se* $\Gamma \vdash M : \tau$ *allora per ogni* Γ- *ambiente* ρ *e per ogni* Γ- *sostituzione* σ*:*

$$\rho \lhd_\Gamma \sigma \Rightarrow [\![\Gamma \vdash M]\!](\rho) \lhd_\tau M[\sigma]$$

Si ricorda che $\rho \lhd_\Gamma \sigma$ *significa che* $\rho(x) \lhd_{\Gamma(x)} \sigma(x)$ *è valido per ogni* $x \in dom(\Gamma)$*, ed inoltre che* $M[\sigma]$ *è il termine risultante dalla simultanea sostituzione di* $\sigma(x)$ *per* x *in* M*, per ogni* $x \in dom(\Gamma)$*.* $\Diamond$

È da notare che la proprietà fondamentale di $\lhd_\tau$ diventa, nel caso in cui $\Gamma = \emptyset$:

$$[\![M]\!] \lhd_\tau M$$

per tutti i tipi τ e per tutti i termini chiusi $M : \tau$.

Ora si può completare la dimostrazione della proprietà dell'*adeguatezza*.

Dimostrazione adeguatezza. Bisogna dimostrare che: $[\![M]\!] = [\![V]\!] \Rightarrow M \Downarrow_\tau V \quad (\tau \in \{nat, bool\})$

$(\tau = nat)$

$V = \mathtt{succ}^n(\mathtt{0})$ per qualche $n \in \mathbb{N}$

$$\begin{array}{l} [\![M]\!] = [\![\mathtt{succ}^n(\mathtt{0})]\!] \\ \Rightarrow n = [\![M]\!] \lhd_\tau M \quad \text{(proprietà fondamentale)} \\ \Rightarrow M \Downarrow \mathtt{succ}^n(\mathtt{0}) \quad \text{(definizione di} \lhd_{nat}) \end{array}$$

($\tau = bool$)

Analoga.

□

11.3 Dimostrazione proprietà fondamentale di $\lhd$

Per la dimostrazione della proprietà fondamentale di $\lhd$ si ha bisogno delle seguenti proprietà sull'approssimazione formale delle relazioni.

Lemma 11.3.1 (Proprietà approssimazione formale)**.**

1. $\forall M \in PCF_\tau \;\; \bot \lhd_\tau M$
2. $\forall\, M \in PCF_\tau$, $\{d \mid d \lhd_\tau M\}$ *è un sottoinsieme del dominio* $[\![\tau]\!]$ *chiuso per catene. Quindi attraverso la* 1 *è un sottoinsieme ammissibile.*
3. $d_2 \sqsubseteq d_1,\; d_1 \lhd_\tau M_1 \;\wedge\; \forall V\,(M_1 \Downarrow_\tau V \Rightarrow M_2 \Downarrow_\tau V) \;\Rightarrow\; d_2 \lhd_\tau M_2$

◊

Dimostrazione. Ognuna di queste proprietà è dimostrabile per induzione sulla struttura di τ, utilizzando le definizioni di $\lhd_\tau$ e di $\Downarrow_\tau$. □

Dimostrazione proprietà fondamentale di $\lhd$ *(prop. 11.2.1).* Per induzione sulle regole di tipaggio $\Gamma \vdash M : \tau$. Si definisce:

$$\Phi(\Gamma, M, \tau) \overset{def}{\Leftrightarrow} \Gamma \vdash M : \tau \;\wedge\; \forall \rho, \sigma\,(\rho \lhd_\Gamma \sigma \Rightarrow [\![\Gamma \vdash M]\!](\rho) \lhd_\tau M[\sigma])$$

Ora, è sufficiente verificare che Φ è chiuso sotto gli assiomi e le regole fornite per le regole di tipaggio.

($:_0$)

$\Phi(\Gamma, \mathtt{0}, nat)$ è vero perché $0 \lhd_{nat} \mathtt{0}$.

($:_{succ}$)

Si deve provare che $\Phi(\Gamma, \mathtt{0}, nat) \Rightarrow \Phi(\Gamma, \mathtt{succ}(M), nat)$. Si verifica facilmente che :

$$d \lhd_{nat} M \Rightarrow s_{\perp}(d) \lhd_{nat} \mathtt{succ}(M)$$

dove $s_{\perp} : \mathbb{N}_{\perp} \to \mathbb{N}_{\perp}$ è la funzione continua utilizzata per descrivere la denotazione del termine successore, $\mathtt{succ}(M)$.

$(:_{pred})$ e $(:_{zero})$

Simili al precedente.

$(:_{bool})$

$\Phi(\Gamma, \mathtt{true}, bool) \Leftarrow true \lhd_{bool} \mathtt{true}$. Analogamente per $\Phi(\Gamma, \mathtt{false}, bool)$.

$(:_{if})$

È sufficiente verificare che $d_1 \lhd_{bool} M_1, d_2 \lhd_{\tau} M_2$ e $d_3 \lhd_{\tau} M_3 \Rightarrow$

$$if(d_1, (d_2, d_3)) \lhd_{\tau} \mathtt{if}\ M_1\ \mathtt{then}\ M_2\ \mathtt{else}\ M_3 \tag{11.1}$$

dove if è la funzione continua : $\mathbb{B}_{\perp} \times (\llbracket\tau\rrbracket \times \llbracket\tau\rrbracket) \to \llbracket\tau\rrbracket$ della proposizione 6.2.5. Se $d_1 = \perp \in \mathbb{B}_{\perp}$ allora $if(d_1, (d_2, d_3)) = \perp$ e la (11.1) è vera per il punto 1 del lemma 11.3.1. Così si può assumere $d_1 \neq \perp$, dove $d_1 = true$ oppure $d_1 = false$. Si considera il caso $d_1 = true$; per $d_1 = false$ la dimostrazione è analoga.

Essendo $true = d_1 \lhd_{bool} M_1$, per la definizione di $\lhd_{bool}$ si ha che $M_1 \Downarrow_{bool} \mathtt{true}$. Segue dalla regola $(\Downarrow_{if1})$:

$$\forall V\,(M_2 \Downarrow_{\tau} V \Rightarrow \mathtt{if}\ M_1\ \mathtt{then}\ M_2\ \mathtt{else}\ M_3 \Downarrow_{\tau} V)$$

Così per il punto 3 del lemma 11.3.1 applicato a $d_2 \lhd_{\tau} M_2$:

$$d_2 \lhd_{\tau} \mathtt{if}\ M_1\ \mathtt{then}\ M_2\ \mathtt{else}\ M_3$$

ed allora poiché $d_2 = if\ \ (true, (d_2, d_3)) = if(d_1, (d_2, d_3))$ si verifica la proprietà (11.1), come richiesto.

$(:_{var})$

$\Phi(\Gamma, x, \Gamma(x))$ è vero poiché se $\rho \lhd_{\Gamma} \sigma$ allora $\forall x \in dom(\Gamma)$ si ha che: $\llbracket\Gamma \vdash x\rrbracket(\rho) \stackrel{def}{=} \rho(x) \lhd_{\Gamma(x)} \sigma(x) \stackrel{def}{=} x[\sigma]$.

$(:_{fn})$

Si supponga che $\Phi(\Gamma[x \mapsto \tau], M, \tau')$ e $\rho \lhd_{\Gamma} \sigma$ siano vere. La tesi è $\llbracket\Gamma \vdash \mathtt{fn}\ x : \tau.M\rrbracket(\rho) \lhd_{\tau \to \tau'} (\mathtt{fn}\ x : \tau.M)[\sigma]$, e.g. $d \lhd_{\tau} N$ implica

$$[\![\Gamma \vdash \texttt{fn}\ x : \tau.M]\!]\,(\rho)\,(d) \lhd_{\tau'} ((\texttt{fn}\ x : \tau.M)\,[\sigma])\,N \tag{11.2}$$

Applicando le regole della semantica denotazionale:

$$[\![\Gamma \vdash \texttt{fn}\ x : \tau.M]\!]\,(\rho)\,(d) = [\![\Gamma\,[x \mapsto \tau] \vdash M]\!]\,(\rho\,[x \mapsto d]) \tag{11.3}$$

Poiché $(\texttt{fn}\ x : \tau.M)\,[\sigma] = \texttt{fn}\ x : \tau.M\,[\sigma]$ e $(M\,[\sigma])\,[N/x] = M\,[\sigma\,[x \mapsto N]]$, per $(:_{cbn})$:

$$\forall V\,(M\,[\sigma\,[x \mapsto N]] \Downarrow_{\tau'} V) \Rightarrow ((\texttt{fn}\ x : \tau.M)\,[\sigma])\,N \Downarrow_{\tau'} V \tag{11.4}$$

Poiché $\rho \lhd_\Gamma \sigma$ e $d \lhd_\tau N$, si ha che $\rho\,[x \mapsto d] \lhd_{\Gamma[x \mapsto \tau]} \sigma\,[x \mapsto N]$; quindi con $\Phi\,(\Gamma\,[x \mapsto \tau], M, \tau')$:

$$[\![\Gamma\,[x \mapsto \tau] \vdash M]\!]\,(\rho\,[x \mapsto d]) \lhd_{\tau'} M\,[\sigma\,[x \mapsto N]]$$

Allora la (11.2) segue da questa per l'applicazione del punto 3 del lemma 11.3.1 alla (11.3) ed alla (11.4).

$(:_{app})$

È sufficiente far vedere che $d_1 \lhd_{\tau \to \tau'} M_1 \ \wedge\ d_2 \lhd_\tau M_2 \Rightarrow d_1\,(d_2) \lhd_{\tau'} M_1 M_2$. Ciò segue immediatamente dalla definizione di $\lhd_{\tau \to \tau'}$.

$(:_{fix})$

Si supponga $\Phi\,(\Gamma, M, \tau \to \tau)$ vera. Per ogni $\rho \lhd_\Gamma \sigma$, bisogna provare che:

$$[\![\Gamma \vdash \texttt{fix}\,(M)]\!]\,(\rho) \lhd_\tau \texttt{fix}\,(M)\,[\sigma] \tag{11.5}$$

Applicando le regole della semantica denotazionale si ha che $[\![\Gamma \vdash \texttt{fix}\,(M)]\!]\,(\rho) = fix\,(f)$, dove $f \stackrel{def}{=} [\![\Gamma \vdash M]\!]\,(\rho)$. Per il punto 2 del lemma 11.3.1:

$$S \stackrel{def}{=} \{d \mid d \lhd_\tau \texttt{fix}\,(M)\,[\sigma]\}$$

ed è un sottoinsieme ammissibile del dominio $[\![\tau]\!]$. Così per il principio di induzione di Scott per verificare la proprietà (11.5) bisogna provare:

$$\forall d \in [\![\tau]\!]\,(d \in S \Rightarrow f\,(d) \in S)$$

Quindi, $\rho \lhd_\Gamma \sigma$, da $\Phi\,(\Gamma, M, \tau \to \tau)$ e dalla definizione della f si ha $f \lhd_{\tau \to \tau} M\,[\sigma]$. Così se $d \in S$, i.e. $d \lhd_\tau \texttt{fix}\,(M)\,[\sigma]$, allora per definizione di $\lhd_{\tau \to \tau}$, questo è il caso in cui:

$$f\,(d) \lhd_\tau (M\,[\sigma])\,(\texttt{fix}\,(M)\,[\sigma]) \tag{11.6}$$

La regola $(:\Downarrow_{fix})$ implica che:

$$\forall V\,((M\,[\sigma])\,(\mathtt{fix}\,(M)\,[\sigma])\Downarrow_\tau V \Rightarrow \mathtt{fix}\,(M)\,[\sigma]\Downarrow_\tau V) \tag{11.7}$$

Allora applicando il punto 3 del lemma 11.3.1 alle proprietà (11.6) e (11.7), si ottiene $f\,(d)\lhd_\tau fix\,(M)\,[\sigma]$, i.e. $f\,(d)\in S$, come richiesto. □

11.4 Estensionalità

Vediamo come l'equivalenza contestuale può diventare un ordine contestuale.

Definizione 11.4.1 (Preordine contestuale)**.** *Siano i termini M_1 e M_2, il tipo τ e l'ambiente Γ, la relazione :*

$$\Gamma\vdash M_1\leq_{ctx} M_2:\tau$$

è vera se e solo se:

- $\Gamma\vdash M_1:\tau$ *e* $\Gamma\vdash M_2:\tau$
- *per ogni contesto C tale che $C\,[M_1]$ e $C\,[M_2]$ sono termini chiusi di tipo γ (dove $\gamma\in\{nat,bool\}$) e per ogni valore $V:\gamma$*

$$C\,[M_1]\Downarrow_\gamma V \Rightarrow C\,[M_2]\Downarrow_\gamma V$$

◊

Chiaramente:

$$\Gamma\vdash M_1\cong_{ctx} M_2:\tau \;\Leftrightarrow\; (\Gamma\vdash M_1\leq_{ctx} M_2:\tau \;\wedge\; \Gamma\vdash M_2\leq_{ctx} M_1:\tau)$$

Come notazione, si scrive $M_1\leq_{ctx} M_2:\tau$ per $\emptyset\vdash M_1\leq_{ctx} M_2:\tau$ qualora M_1 e M_2 siano termini chiusi.

La relazione formale di approssimazione $\lhd_\tau$ caratterizza il preordine contestuale tra termini chiusi, come si vede nella prossima proposizione.

Proposizione 11.4.2 (Preordine contestuale da approssimazioni formali)**.** *Per ogni tipo τ e per tutti i termini chiusi M_1 , $M_2\in PCF_\tau$*

$$M_1\leq_{ctx} M_2:\tau \;\Leftrightarrow\; [\![M_1]\!]\lhd_\tau M_2$$

◊

Dimostrazione. ($\Leftarrow$) Si deve dimostrare che per i termini chiusi $M_1, M_2 \in PCF_\tau$, $M_1 \leq_{ctx} M_2 : \tau$ è vero se e solo se per ogni $M \in PCF_{\tau\to bool}$:

$$MM_1 \Downarrow_{bool} \texttt{true} \Rightarrow MM_2 \Downarrow_{bool} \texttt{true}$$

Se $[\![M_1]\!] \lhd_\tau M_2$ allora per un qualunque $M \in PCF_{\tau\to bool}$ per la proprietà fondamentale di $\lhd$ si ha che $[\![M]\!] \lhd_{\tau\to bool} M$, la definizione di $\lhd_{\tau\to bool}$ implica che:

$$[\![MM_1]\!] = [\![M]\!]\,([\![M_1]\!]) \lhd_{bool} MM_2 \tag{11.8}$$

Così, se $MM_1 \Downarrow_{bool} \texttt{true}$, allora $[\![MM_1]\!] = true$ (per la proprietà di correttezza) e quindi per la definizione di $\lhd_{bool}$ per la proprietà (11.8) si ha che $MM_2 \Downarrow_{bool} \texttt{true}$. Così, utilizzando la caratterizzazione di $\leq_{ctx}$ menzionata precedentemente, si ha $M_1 \leq_{ctx} M_2 : \tau$.

($\Rightarrow$) Bisogna provare che:

$$(d \lhd_\tau M_1 \ \wedge \ M_1 \leq_{ctx} M_2 : \tau) \Rightarrow d \lhd_\tau M_2 \tag{11.9}$$

Se $M_1 \leq_{ctx} M_2 : \tau$, essendo $[\![M_1]\!] \lhd_\tau M_1$ (per la proprietà fondamentale), la proprietà (11.9) implica $[\![M_1]\!] \lhd_\tau M_2$. La proprietà (11.9) segue dall'induzione sulla struttura del tipo τ, utilizzando la verifica della proprietà di $\leq_{ctx}$:

- se $\tau \in \{nat, bool\}$ allora $M_1 \leq_{ctx} M_2 : \tau$ implica $\forall V : \tau\,(M_1 \Downarrow_\tau V \Rightarrow M_2 \Downarrow_\tau V)$
- se $M_1 \leq_{ctx} M_2 : \tau \to \tau'$ allora $M_1 M \leq_{ctx} M_2 M : \tau'$ per ogni $M : \tau$

□

La doppia implicazione permette di trasferire le proprietà di estensionalità utilizzate nell'ordine parziale di un dominio $\sqsubseteq$ verso il preordine contestuale, mostrato nella prossima proposizione.

Proposizione 11.4.3 (Estensionalità di $\leq_{ctx}$)**.**

- *Sia $\tau \in \{bool, nat\}$, $M_1 \leq_{ctx} M_2 : \tau$ se e solo se:*

$$\forall V : \tau\,(M_1 \Downarrow_\tau V \Rightarrow M_2 \Downarrow_\tau V)$$

- *Sia il tipo $\tau \to \tau'$, $M_1 \leq_{ctx} M_2 : \tau \to \tau'$ se e solo se:*

$$\forall M : \tau\,(M_1 M \leq_{ctx} M_2 M : \tau')$$

◊

Dimostrazione.

($\Rightarrow$) È una conseguenza della definizione di $\leq_{ctx}$.

($\Leftarrow$)

- $\tau \in \{bool, nat\}$

$$\begin{array}{ll} [\![M_1]\!] = [\![V]\!] \Rightarrow M_1 \Downarrow_\tau V & \text{(adeguatezza)} \\ \Rightarrow M_2 \Downarrow_\tau V & \text{(assunzione)} \end{array}$$

  e quindi $[\![M_1]\!] \lhd_\tau M_2$ per definizione di $\lhd$ su questi tipi. Ora si applica la proposizione sul preordine contestuale da approssimazioni formali.

- tipo $\tau \to \tau'$

$$\begin{array}{rcll} d \lhd_\tau M & \Rightarrow & [\![M_1]\!]\,(d) \lhd_{\tau'} M_1 M & ([\![M_1]\!] \lhd_\tau M_1) \\ & \Rightarrow & [\![M_1]\!]\,(d) \lhd_{\tau'} M_2 M & ((11.9) \text{ e } M_1 M \leq_{ctx} M_2 M : \tau' \text{ per assunzione}) \end{array}$$

  quindi $[\![M_1]\!] \lhd_{\tau \to \tau'} M_2$ per definizione di $\lhd$ di tipo $\tau \to \tau'$. Così, ancora una volta si può applicare la proposizione sul preordine contestuale da approssimazioni formali per terminare la dimostrazione.

□

Capitolo 12

Astrazione completa

12.1 Fallimento dell'astrazione completa

Come si è visto nel teorema 9.5.7, la proprietà di adeguatezza implica che l'equivalenza contestuale di due termini può essere provata mostrando che hanno uguale denotazione: $[\![M_1]\!] = [\![M_2]\!] \in [\![\tau]\!] \Rightarrow M_1 \cong_{ctx} M_2 : \tau$. Sfortunatamente il viceversa è falso: esistono equivalenze contestuali di termini con denotazioni diverse. In generale, si dice che una semantica denotazionale è *completamente astratta* se l'equivalenza contestuale coincide con l'uguaglianza della denotazione. La semantica denotazionale di PCF, utilizzando dominî e funzioni continue, non è completamente astratta. L'esempio classico che dimostra questo fallimento è dovuto a Plotkin (1977) e si basa sull' "or-parallelo" (o "por") e sull'"or-sequenziale" (o "orelse").

Definizione 12.1.1 (Funzione di or-parallelo)**.** *La funzione continua por* : $\mathbb{B}_\perp \to (\mathbb{B}_\perp \to \mathbb{B}_\perp)$ *è così definita:*

por	*true*	*false*	$\perp$
true	*true*	*true*	*true*
false	*true*	*false*	$\perp$
$\perp$	*true*	$\perp$	$\perp$

$\Diamond$

Definizione 12.1.2 (Funzione di or-sequenziale)**.** *La funzione continua orelse* : $\mathbb{B}_\perp \to (\mathbb{B}_\perp \to \mathbb{B}_\perp)$ *è così definita:*

È la denotazione del termine PCF:

$$\texttt{fn}\ x : \mathit{bool}.\ \texttt{fn}\ x' : \mathit{bool}.\ \texttt{if}\ x\ \texttt{then}\ \texttt{true}\ \texttt{else}\ x'$$

$orelse$	$true$	$false$	$\bot$
$true$	$true$	$true$	$true$
$false$	$true$	$false$	$\bot$
$\bot$	$\bot$	$\bot$	$\bot$

di tipo $bool \to (bool \to bool)$. $\Diamond$

Si confronti la funzione *por* con la funzione $orelse$. Entrambe le funzioni forniscono l'usuale funzione booleana "or" se ristrette a $\{true, false\}$, ma differiscono nel loro comportamento con l'argomento $\bot$, il quale rappresenta la "non terminazione". Si osservi che $por\,(d_1, d_2) = true$ se d_1 o d_2 sono veri (i.e. anche se l'altro argomento è $\bot$); invece $orelse\,(d_1, d_2) = true$ implica che $d_1 \neq \bot$.

Come osservato in 12.1.2, la funzione $orelse$ può essere definita in PCF i.e. esiste un termine chiuso $M : bool \to (bool \to bool)$ tale che $[\![M]\!] = orelse$. Il termine M esegue il test sul primo argomento[1], se è `true` restituisce `true` (e ignora il secondo argomento), altrimenti restituisce il secondo argomento.

Per quanto riguarda *por*, si consideri la seguente proposizione.

Proposizione 12.1.3 (Indefinibilità dell'or-parallelo)**.** *Non esiste alcun termine chiuso* $P : bool \to (bool \to bool)$ *che soddisfi* $[\![P]\!] = por$. $\Diamond$

Non se ne darà una dimostrazione qui. [Plo77] prova mediante un *Activity Lemma*, ma ci sono approcci alternativi che utilizzano funzioni continue "stabili" ([Gun92]) o utilizzano "relazioni logiche sequenziali" ([Sie92]). L'idea principale è che la valutazione proceda in modo sequenziale. Segue che per ogni P, la valutazione di PM_1M_2 deve implicare in qualche momento la valutazione completa di M_1 o M_2 (P non può ignorare i sui argomenti se restituisce `true` in alcuni casi e `false` in altri); mentre un algoritmo per calcolare la funzione *por* su una coppia di argomenti, deve calcolare i valori di tali argomenti in "parallelo", nel caso uno diverga mentre l'altro produce il valore *true*.

Si può sfruttare l'indefinibilità della funzione *por* in PCF per creare una coppia di termini chiusi contestualmente equivalenti con denotazioni diverse.

Proposizione 12.1.4 (Fallimento della proprietà di astrazione completa)**.** *Per i=1,2 si definisce:*

$T_i \stackrel{def}{=}$ `fn` $f : bool \to (bool \to bool)$
`if` $(f$ `true` $\Omega)$ `then`
`if` $(f\ \Omega$ `true`$)$ `then`
`if` $(f$ `false` `false`$)$ `then` Ω `else` B_i

[1] Quindi diverge se il primo argomento diverge.

$$\begin{array}{l} \quad \texttt{else}\ \Omega \\ \texttt{else}\ \Omega \end{array}$$

dove $B_1 \stackrel{def}{=} \texttt{true}, B_2 \stackrel{def}{=} \texttt{false}$.

$\Omega \stackrel{def}{=} \texttt{fix}(\texttt{fn}\ x : bool.x)$.

Allora

$$T_1 \cong_{ctx} T_2 : (bool \to (bool \to bool)) \to bool$$
$$[\![T_1]\!] \neq [\![T_2]\!] \in (\mathbb{B}_\perp \to (\mathbb{B}_\perp \to \mathbb{B}_\perp)) \to \mathbb{B}_\perp$$

$\Diamond$

Dimostrazione. Per la definizione di *por* e per la definizione di $[\![-]\!]$ in 9.5, si ha che:

$$[\![T_i]\!]\,(por) = \begin{cases} true & se\ i = 1 \\ false & se\ i = 2 \end{cases}$$

Quindi $[\![T_1]\!](por) \neq [\![T_2]\!](por)$, perciò $[\![T_1]\!] \neq [\![T_2]\!]$.

Per verificare che $T_1 \cong_{ctx} T_2 : (bool \to (bool \to bool)) \to bool$ si utilizzano le proprietà di estensionalità di $\leq_{ctx}$ (11.4.3).

Si deve mostrare che per ogni $M : bool \to (bool \to bool)$ e $V \in \{\texttt{true}, \texttt{false}\}$:

$$T_1 M \Downarrow_{bool} V \Leftrightarrow T_2 M \Downarrow_{bool} V \tag{12.1}$$

Ma la definizione di T_i è tale che $T_i M \Downarrow_{bool} V$ vale se:

$$M\ \texttt{true}\ \Omega \Downarrow_{bool} \texttt{true}, M\ \Omega\ \texttt{true} \Downarrow_{bool} \texttt{true}, M\ \texttt{false}\ \texttt{false} \Downarrow_{bool} \texttt{false}$$

Per la proprietà di correttezza si ha che:

$$[\![M]\!]\,(true)\,(\perp) = true, [\![M]\!]\,(\perp)\,(true) = true, [\![M]\!]\,(false)\,(false) = false$$

Poiché vale $[\![\Omega]\!] = \perp$.

Segue che la funzione continua $[\![M]\!] : (\mathbb{B}_\perp \times \mathbb{B}_\perp) \to \mathbb{B}_\perp$ coincide con la funzione *por*.

Ma ciò è impossibile per la proposizione 12.1.3. Quindi la (12.1) è soddisfatta per ogni M, e T_1 e T_2 sono contestualmente equivalenti. □

Un risultato interessante è che Plotkin dimostra che questa è l'unica cosa non esprimibile, quindi se si aggiunge una definizione per questo si ha che tutto è esprimibile. Per approfondimenti si rimanda a [Plo77].

Il fatto che la sequenza crei problemi è un campanello d'allarme: il calcolo sequenziale oltre che meno efficiente del parallelo ha anche questo tipo di problemi; per questo motivo è importante studiare il calcolo parallelo (appuntamento al corso *Teoria della Concorrenza*!).

Parte IV

Esercizi

Capitolo 13

Esercizi

13.1 Esercizio sui dominî di funzione

Prendiamo in considerazione tutte le funzioni parziali $f : X \nrightarrow Y$, con dominio di definizione $dom(f) \subseteq X$ che mappano valori in Y. Definiamo la relazione di ordine come segue:$f \sqsubseteq g$ se e solo se $dom(f) \subseteq dom(g)$ e $\forall x \in dom(f)$, $f(x) = g(x)$. Mostriamo che questa è una relazione d'ordine parziale.

Dimostrazione. La relazione d'ordine sopra definita è una relazione d'ordine parziale:

Riflessiva:

$f \sqsubseteq f$ e $dom(f) \subseteq dom(f)$, ma allora banalmente $\forall x \in dom(f)\ f(x) = f(x)$.

Transitiva:

$f \sqsubseteq g$ e $g \sqsubseteq h$ con $dom(f) \subseteq dom(g)$, $dom(g) \subseteq dom(h)$ e $\forall x \in dom(f)$ $\forall y \in dom(g)\ f(x) = g(x)$ e $g(y) = h(y)$, ma allora $dom(f) \subseteq dom(g) \subseteq dom(h)$ ed inoltre $\forall x \in dom(f)\ f(x) = h(x)$ quindi $f \sqsubseteq h$.

Antisimmetrica:

$f \sqsubseteq g$ e $g \sqsubseteq f$ con $dom(f) \subseteq dom(g)$, $dom(g) \subseteq dom(f)$ e $\forall x \in dom(f) = dom(g)\ f(x) = g(x)$ e $g(y) = h(y)$. Poiché sono definite sullo stesso dominio e prendono gli stessi valori allora $f = g$.

lub:

Sia $f_0 \sqsubseteq f_1 \sqsubseteq f_2$.... la catena considerata, e sia f la funzione parziale con $dom(f) = \bigcup_{n\geq 0} dom(f_i)$:

$$f(x) = \begin{cases} f_i(x) & \text{se } x \in dom(f_i) \text{ per qualche } i \\ \bot & \text{altrimenti} \end{cases} \tag{13.1}$$

La funzione è ben definita poiché è concorde su tutti i valori. Per dimostrare che $f(x)$ sia un lub mostriamo prima che è un *confine superiore*.

La catena delle funzioni parziali sarà così formata $f_0 \sqsubseteq f_1 \sqsubseteq f_2 \ldots$, quindi per come è stata definita la relazione di ordine parziale $dom(f_0) \subseteq dom(f_1) \subseteq dom(f_2) \ldots$ affinché $f(x)$ sia un confine superiore deve valere che : $\forall i \in \mathbb{N} \;\; f_i(x) \sqsubseteq f(x)$, ma questo è vero poichè $\forall i \;\; dom(f_i) \subseteq dom(f) = \bigcup_{n\geq 0} dom(f_i)$ ed inoltre $\forall x \in dom(f_i) \;\; f_i(x) = f(x)$ per definizione della funzione $f(x)$.

Mostriamo ora che è anche il minimo confine superiore.

Prendiamo in considerazione un generico confine superiore f_p per la catena di funzioni parziali, e mostriamo che $f(x)$ lo precede. Quindi, $\forall i \in \mathbb{N} \;\; f_i \sqsubseteq f_p \Rightarrow dom(f) = \bigcup_{n\geq 0} dom(f_i) \subseteq dom(f_p)$ e poiché $\forall x \in \bigcup_{n\geq 0} dom(f_i) = dom(f)$ abbiamo che $f(x) = f_i(x) = f_p(x)$ allora $f \sqsubseteq f_p$, f è quindi un lub per la catena delle funzioni parziali.

L'elemento minimo appartiene ad f:

L'elemento minimo che indichiamo con $\bot$ è compreso nella funzione poiché $dom(\bot) = \emptyset$ e quindi $\forall i \in \mathbb{N} \;\; dom(\bot) \subseteq dom(f_i)$ e $\forall x \in dom(\bot) \;\; f_i(x) = \bot$, $\forall i \in \mathbb{N} \;\; \bot \sqsubseteq f_i$. □

13.2 Esercizio sulla costruzione dei dominî

Siano X e Y due insiemi ed $X_\bot$ e $Y_\bot$ i corrispondenti dominî piatti. Bisogna mostrare che la funzione $f : X_\bot \to Y_\bot$ è continua se e solo se vale una delle seguenti affermazioni

- f è esatta i.e. $f(\bot) = \bot$
- f è costante i.e. $\forall x \in X. f(x) = f(\bot)$

Dimostrazione. ($\Rightarrow$) Supponiamo che f sia continua e mostriamo che vale una delle due affermazioni. Poiché f è continua: $\forall x_1, x_2 \in X \;\; x_1 \sqsubseteq x_2 \Rightarrow f(x_1) \sqsubseteq f(x_2)$. Supponiamo che $x_1 = \bot$ allora $\bot \sqsubseteq x_2 \Rightarrow f(\bot) \sqsubseteq f(x_2)$, ricordando che siamo nei dominî piatti ci sono solo due possibilità affinchè $f(\bot) \sqsubseteq f(x_2)$:

- $f(\bot) = \bot$ f è esatta;

- $f(\perp) = f(x_2)$ f è costante.

Nel caso in cui $x_1 \neq \perp$ e $x_2 \neq \perp$ essendo nei dominî piatti è mantenuta la monotonia.

($\Leftarrow$) Supponiamo ora che f sia esatta, mostriamo che è continua sui dominî piatti. Verifichiamo prima la monotonia. Poiché $f(\perp) = \perp$, $\forall x_1, x_2 \in X_\perp$, se $x_1 \sqsubseteq x_2$ allora $x_1 = \perp \Rightarrow f(\perp) = \perp \sqsubseteq f(x_2)$, altrimenti ricordando che siamo nei dominî piatti $f(x_1) \sqsubseteq f(x_2)$ perché $x_1 = x_2$, quindi se f è esatta è monotona, ma essendo nei dominî piatti è anche banalmente continua poiché le catene di elementi sono finite.

Mostriamo che f è continua anche se costante sui dominî piatti,verifichiamo la monotonia $\forall x_1, x_2 \in X\ x_1 \sqsubseteq x_2 \Rightarrow f(x_1) \sqsubseteq f(x_2)$; quanto detto vale poiché essendo f costante $f(x_1) = f(x_2) = \perp$ e quindi $f(x_1) \sqsubseteq f(x_2)$. Per quanto riguarda la conservazione del lub, vengono mappati elementi in catene finite, quindi se la funzione è monotona nei dominî piatti è anche continua. □

13.3 Esercizio sul PCF

Per ogni tipo τ e per ogni coppia di termini chiusi M_1,$M_2 \in PCF_\tau$ mostrare che :

$$\forall V : \tau(M_1 \Downarrow_\tau V \Leftrightarrow M_2 \Downarrow_\tau V) \Rightarrow M_1 \cong_{ctx} M_2 : \tau$$

Dimostrazione. Per 11.4:

$$\forall V : \tau(M_1 \Downarrow_\tau V \Leftrightarrow M_2 \Downarrow_\tau V) \Rightarrow M_1 \cong_{ctx} M_2 : \tau \Leftrightarrow (M_1 \leq_{ctx} M_2 : \tau \wedge M_2 \leq_{ctx} M_1 : \tau) \tag{13.2}$$

Applicando la definizione 11.4.1:

$$(M_1 \leq_{ctx} M_2 : \tau \wedge M_2 \leq_{ctx} M_1 : \tau) \Leftrightarrow ([\![M_1]\!] \lhd_\tau M_2 \wedge [\![M_2]\!] \lhd_\tau M_1) \tag{13.3}$$

Ma per (11.1.2) possiamo dire che $[\![M_1]\!] = [\![V]\!] = [\![M_2]\!]$. Quindi abbiamo mostrato la formula. □

13.4 Esercizio valutazione PCF

Dati i seguenti termini:

$plus \stackrel{def}{=} \texttt{fix}(\texttt{fn}\ \ p : nat \to (nat \to nat).\texttt{fn}\ \ x : nat.\texttt{fn}\ \ y : \texttt{nat}.$
$\quad \texttt{if}\ \ \texttt{zero}(y)\ \ \texttt{then}\ \ x\ \ \texttt{else}\ \ \texttt{succ}(p\, x\ \ \texttt{pred}(y)))$

$times \stackrel{def}{=} \texttt{fix}(\texttt{fn}\ \ t : nat \to (nat \to nat).\texttt{fn}\ \ x : nat.\texttt{fn}\ \ y : nat.$
$\quad \texttt{if}\ \ \texttt{zero}(y)\ \ \texttt{then}\ \ \texttt{0}\ \ \texttt{else}\ \ plus(t\, x\ \ \texttt{pred}(y))x)$

$fact \stackrel{def}{=} \texttt{fix}(\texttt{fn}\ \ f : nat \to nat.\texttt{fn}\ \ x : nat.$
$\quad \texttt{if}\ \ \texttt{zero}(x)\ \ \texttt{then}\ \ \texttt{succ(0)}\ \ \texttt{else}\ \ times\, x(f\ \ \texttt{pred}(x)))$

Mostrare per induzione su $n \in \mathbb{N}$ che per ogni $m \in \mathbb{N}$ si ha:

$$plus \quad \texttt{succ}^m(\texttt{0})\texttt{succ}^n(\texttt{0}) \Downarrow_{nat} \texttt{succ}^{m+n}(\texttt{0}) \tag{13.4}$$

$$times \quad \texttt{succ}^m(\texttt{0})\texttt{succ}^n(\texttt{0}) \Downarrow_{nat} \texttt{succ}^{m*n}(\texttt{0}) \tag{13.5}$$

$$fact \quad \texttt{succ}^n(\texttt{0}) \Downarrow_{nat} \texttt{succ}^{n!}(\texttt{0}) \tag{13.6}$$

Per risolvere l'esercizio utilizzeremo gli assiomi di valutazione (9.4.2).

Dimostrazione. L'esercizio richiede di verificare che per i tre termini introdotti precedentemente valgono le proprietà (13.4), (13.5) e (13.6).

Innanzitutto, bisogna verificare che i termini $plus$, $times$ e $fact$, svolgano, rispettivamente, le operazioni di somma, prodotto e fattoriale di termini **PCF**.

La verifica è per induzione su $n \in \mathbb{N}$ utilizzando (9.4.2).

Verifichiamo la proprietà (13.4) per il termine $plus$.

Passo base $(n = 0)$.

$plus\ \ \texttt{succ}^m(\texttt{0})\ \ \texttt{succ}^{\texttt{0}}(\texttt{0})$
$\Downarrow_{nat}\ \ \texttt{fix}(\texttt{fn}\ \ p : nat \to (nat \to nat).\texttt{fn}\ \ x : nat.\texttt{fn}\ \ y : nat.$
$\qquad \texttt{if}\ \ \texttt{zero}(y)\ \ \texttt{then}\ \ x\ \ \texttt{else}\ \ \texttt{succ}(p\, x\ \ \texttt{pred}(y)))\texttt{succ}^m(\texttt{0})\texttt{succ}^{\texttt{0}}(\texttt{0})$
$\Downarrow_{nat}\ \ \texttt{fix}(\texttt{fn}\ \ p : nat \to (nat \to nat).\texttt{fn}\ \ x : nat.\texttt{fn}\ \ y : nat.$
$\qquad \texttt{if}\ \ \texttt{zero}(y)\ \ \texttt{then}\ \ x\ \ \texttt{else}\ \ \texttt{succ}(p\, x\ \ \texttt{pred}(y)))\ \ \texttt{succ}^m(\texttt{0})\texttt{0}$ (def. del tipo nat cf
$\Downarrow_{nat}\ \ \texttt{fix}(\texttt{fn}\ \ p : nat \to (nat \to nat))\ \ \texttt{succ}^m(\texttt{0})$ ($\Downarrow_{zero1}$, $\Downarrow_{if1}$)
$\Downarrow_{nat}\ \ \texttt{fix}(\ \ \texttt{succ}^m(\texttt{0}))$ ($\Downarrow_{cbn}$)
$\Downarrow_{nat}\ \ \textbf{succ}^m(\textbf{0})$ ($\Downarrow_{cbn}$)

Passo induttivo. Si supponga la tesi vera per $n \in \mathbb{N}$, si verifica per $n + 1$:

$plus$ $\texttt{succ}^m(0)\texttt{succ}^{n+1}(0)$
$\Downarrow_{nat}$ $\texttt{fix}(\texttt{fn}\ p : nat \to (nat \to nat).\texttt{fn}\ x : nat.\texttt{fn}\ y : nat.$
$\quad \texttt{if zero}(y)\ \texttt{then}\ x\ \texttt{else succ}(p\, x\ \ \texttt{pred}(y)))\ \texttt{succ}^m(0)\ \texttt{succ}^{n+1}(0)$
$\Downarrow_{nat}$ $\texttt{fix}(\texttt{fn}\ p : nat \to (nat \to nat))\texttt{succ}(p\ \ \texttt{succ}^m(0)\texttt{pred succ}^{n+1}(0))$ $(\Downarrow_{zero2}, \Downarrow_{if2})$
$\Downarrow_{nat}$ $\texttt{fix}(\texttt{fn}\ p : nat \to (nat \to nat))\ \texttt{succ}(p\ \ \texttt{succ}^m(0)\ \ \texttt{succ}^n(0))$ $(\Downarrow_{pred})$
$\Downarrow_{nat}$ $\texttt{fix}(\texttt{fn}\ p : nat \to (nat \to nat))\texttt{succ}(\texttt{succ}^{m+n}(0))$ (IH)
$\Downarrow_{nat}$ $\texttt{fix}(\texttt{fn}\ p : nat \to (nat \to nat))\ \texttt{succ}^{m+n+1}(0)$ (def. del tipo nat cfr. sez.
$\Downarrow_{nat}$ $\texttt{fix}(\texttt{succ}^{m+n+1}0)$ $(\Downarrow_{cbn})$
$\Downarrow_{nat}$ $\texttt{succ}^{m+n+1}(0)$ $(\Downarrow_{cbn})$

Verifichiamo la relazione (13.5) per il termine $times$:

Passo base ($n = 0$).

$times$ $\textbf{succ}^m(\textbf{0})\ \textbf{succ}^{\textbf{0}}(\textbf{0})$
$\Downarrow_{nat}$ $\texttt{fix}(\texttt{fn}\ t : nat \to (nat \to nat).\texttt{fn}\ x : nat.\texttt{fn}\ y : nat.$
$\quad \texttt{if zero}(y)\ \ \texttt{then 0 else}\ plus(t\, x\ \ \texttt{pred}(y))x)\ \texttt{succ}^m(0)\texttt{succ}^0(0)$
$\Downarrow_{nat}$ $\texttt{fix}(\texttt{fn}\ t : nat \to (nat \to nat).\texttt{fn}\ x : nat.\texttt{fn}\ y : nat.$
$\quad \texttt{if zero}(y)\ \texttt{then 0 else}\ plus(t\, x\ \ \texttt{pred}(y))x)\texttt{succ}^m(0)\ \texttt{0}$ (def. del tipo nat cfr. sez. 9.4
$\Downarrow_{nat}$ $\texttt{fix}(\texttt{fn}\ t : nat \to (nat \to nat))\ \texttt{0}$ $(\Downarrow_{zero1}, \Downarrow_{if1})$
$\Downarrow_{nat}$ $\texttt{fix}(0)$ $(\Downarrow_{cbn})$
$\Downarrow_{nat}$ $\texttt{0}$ $(\Downarrow_{cbn})$

Passo induttivo. Si supponga la tesi vera per $n \in \mathbb{N}$, si verifica per $n + 1$:

$times$ $\texttt{succ}^m(0)\texttt{succ}^{n+1}(0)$
$\Downarrow_{nat}$ $\texttt{fix}(\texttt{fn}\ t : nat \to (nat \to nat).\texttt{fn}\ x : nat.\texttt{fn}\ y : nat.$
$\quad \texttt{if zero}(y)\ \ \texttt{then 0 else}\ plus(t\, x\ \ \texttt{pred}(y))x)\ \texttt{succ}^m(0)\ \texttt{succ}^{n+1}(0)$
$\Downarrow_{nat}$ $\texttt{fix}(\texttt{fn}\ t : nat \to (nat \to nat))\ plus(t(\texttt{succ}^m(0)\ \texttt{pred succ}^{n+1}(0))\texttt{succ}^m(0))$ $(\Downarrow_{zero2}, \Downarrow_{if2})$
$\Downarrow_{nat}$ $\texttt{fix}(\texttt{fn}\ t : nat \to (nat \to nat))\ plus(t(\ \texttt{succ}^m(0)\texttt{succ}^n(0))\texttt{succ}^m(0))$ $(\Downarrow_{pred})$
$\Downarrow_{nat}$ $\texttt{fix}(\texttt{fn}\ t : nat \to (nat \to nat))\ plus(\texttt{succ}^{m*n}(0)\texttt{succ}^m(0))$ (IH)
$\Downarrow_{nat}$ $\texttt{fix}(\texttt{fn}\ t : nat \to (nat \to nat))\ \ \texttt{succ}^{m*n+m}(0)$ (def. $plus$)
$\Downarrow_{nat}$ $\texttt{fix}(\texttt{succ}^{m*n+m}(0))$ $(\Downarrow_{cbn})$
$\Downarrow_{nat}$ $\texttt{succ}^{m*n+m}(0)$ $(\Downarrow_{cbn})$
$\Downarrow_{nat}$ $\texttt{succ}^{m*(n+1)}(0)$

Infine, verifichiamo la relazione (13.6) per il termine $fact$:

Passo base ($n = 0$).

$fact$ $\texttt{succ}^0(0)$
$\Downarrow_{nat}$ $\texttt{fix}(\texttt{fn}\ f : nat \to (nat \to nat).\texttt{fn}\ x : nat.$
$\quad \texttt{if zero}(x)\ \ \texttt{then succ}(0)\ \texttt{else}\ times\, x(f\ \ \texttt{pred}(x))\ \texttt{succ}^0(0)$
$\Downarrow_{nat}$ $\texttt{fix}(\texttt{fn}\ f : nat \to (nat \to nat).\texttt{fn}\ x : nat.$
$\quad \texttt{if zero}(x)\ \ \texttt{then succ}(0)\ \texttt{else}\ times\, x(f\ \ \texttt{pred}(x))0$ (def. del tipo nat cfr. sez. 9.4)
$\Downarrow_{nat}$ $\texttt{fix}(\texttt{fn}\ f : nat \to (nat \to nat))\texttt{succ}(0)$ $(\Downarrow_{zero1}, \Downarrow_{if1})$
$\Downarrow_{nat}$ $\texttt{fix}(\texttt{succ}(0))$ $(\Downarrow_{cbn})$
$\Downarrow_{nat}$ $\texttt{succ}(0)$ $(\Downarrow_{cbn})$

Passo induttivo. Si supponga la tesi vera per $n \in \mathbb{N}$, si verifica per $n+1$:

$$
\begin{array}{lll}
fact\ \ \mathtt{succ}^{n+1}(\mathtt{0}) \Downarrow_{nat}\ \ \mathtt{fix}(\mathtt{fn}\ f : nat \to (nat \to nat).\mathtt{fn}\ x : nat. & & \\
\quad \mathtt{if\ zero}(x)\ \ \mathtt{then\ succ}(\mathtt{0})\ \ \mathtt{else}\ times\, x(f\ \ \mathtt{pred}(x))\ \ \mathtt{succ}^{n+1}(\mathtt{0}) & & \\
\Downarrow_{nat}\ \ \mathtt{fix}(\mathtt{fn}\ f : nat \to (nat \to nat))\ times(\ \mathtt{succ}^{n}(\mathtt{0})\ f(\ \mathtt{pred}\ \mathtt{succ}^{n+1}(\mathtt{0}))) & & (\Downarrow_{zero2}, \Downarrow_{if2}) \\
\Downarrow_{nat}\ \ \mathtt{fix}(\mathtt{fn}\ f : nat \to (nat \to nat))\ times(\mathtt{succ}^{n}(\mathtt{0})\ f(\ \mathtt{succ}^{n}(\mathtt{0}))) & & (\Downarrow_{pred}) \\
\Downarrow_{nat}\ \ \mathtt{fix}(\mathtt{fn}\ f : nat \to (nat \to nat))\ times(\mathtt{succ}^{n}(\mathtt{0})\ \mathtt{succ}^{n!}(\mathtt{0})) & & (\mathrm{IH}) \\
\Downarrow_{nat}\ \ \mathtt{fix}(\mathtt{fn}\ f : nat \to (nat \to nat))\mathtt{succ}^{n!*n}(\mathtt{0}) & & (\text{def. } times) \\
\Downarrow_{nat}\ \ \mathtt{fix}(\mathtt{succ}^{n!*n}(\mathtt{0})) & & (\Downarrow_{cbn}) \\
\Downarrow_{nat}\ \ \mathtt{succ}^{n!*n}(\mathtt{0}) & & (\Downarrow_{cbn}) \\
\Downarrow_{nat}\ \ \mathtt{succ}^{(n+1)!}(\mathtt{0}) & &
\end{array}
$$

□

13.5 Esercizio sui dominî

Sia $\{T\}$ un insieme con un unico elemento e $\{T\}_\perp$ il corrispondente dominio piatto. Sia Ω il dominio dei numeri naturali verticali (esempio 8.0.23). Bisogna mostrare che il dominio di funzione $\Omega \to \{T\}_\perp$ è in biiezione con Ω .

Il dominio di funzione $\Omega \to \{T\}_\perp$ è definito nel seguente modo:

$$\Omega \to \{T\}_\perp = \{f \mid f\colon \Omega \to \{T\}_\perp \text{ è una funzione continua}\}$$

L'insieme Ω dei numeri naturali verticali è il seguente:

Dimostrazione. Per mostrare che il dominio di funzione $\Omega \to \{T\}_\perp$ è in biiezione con l'insieme Ω dei numeri naturali verticali proviamo a mettere in relazione gli elementi dei due insiemi.
Per prima cosa bisogna individuare le funzioni appartenenti a $\Omega \to \{T\}_\perp$ che

per definizione sono tutte e sole le funzioni continue da Ω in $\{T\}_\perp$. Tali funzioni saranno le f_s, $s \in \Omega$, definite nel seguente modo:

$$f_s(x) = \begin{cases} T & se \;\; x \geq s \\ \perp & \text{altrimenti} \end{cases}$$

Bisogna verificare che queste sono tutte e sole le funzioni continue da Ω in $\{T\}_\perp$. Infatti le uniche altre funzioni che possiamo costruire da Ω in $\{T\}_\perp$ sono quelle in cui una volta mandato un certo elemento $s \in \Omega$ nell'elemento T rimandiamo uno degli elementi che in Ω stanno al di sopra di s nel $\perp$. Purtroppo, facendo in questo modo perderemmo la monotonia e, quindi, la continuità.

Rimane ora da verificare la corrispondenza biunivoca tra il dominio di funzione $\Omega \to \{T\}_\perp$ e Ω. Ciò, tuttavia, è immediato in quanto dato un elemento $s \in \Omega$ è univocamente determinata la funzione f_s in $\Omega \to \{T\}_\perp$. Analogamente data una f_p in $\Omega \to \{T\}_\perp$ è univocamente determinato l'elemento $p \in \Omega$ ad esso associato. □

Tabella delle abbreviazioni

Sigla	Significato
CPO	Insieme parzialmente ordinato continuo
IH	Ipotesi induttiva
Inf	Massimo comune minorante
IP	Principio di induzione
Lub	Least upper bound (minimo confine superiore o minimo comune maggiorante)
PCF	Programming Computable Function
Poset	Insieme parzialmente ordinato
Sup	Minimo comune maggiorante

Bibliografia

[Ben98] Ben-Ari M., *Logica matematica per l'informatica*, UTET Libreria, 1998.

[Gal06] *Institue and Museum of the History of the Science*, web site. `http://galileo.imss.firenze.it`

[Gun92] Gunter C., *Semantics of Programming Languages: Structures and Techniques*, MIT Press, Foundations of Computing, 1992.

[Lab00] Labella A., *Dispense del corso "Matematiche Complementari"*, 2000.

[Lab05] Labella A., *Dispense del corso "Semantica dei Linguaggi di Programmazione"*, 2005.

[Mul87] Mulmuley K., *Full abstraction and Semantic Equivalence*, MIT Press, 1987.

[Plo77] Plotkin G., *LCF Considered as a Programming Language*, Theoretical Computer Science 5, 1977.

[Sco93] Scott D., *A Type-theoretical Alternative to ISWIM, CUCH, OWHY*, Theoretical Computer Science 121, 1993[1].

[Sie92] Sieber K., *Reasoning about sequential functions via logical relations*, Applications of Categories in Computer Science, Proceedings LMS Symposium, 1991.

[Wik06] *Wikipedia*, sito web. `http://it.wikipedia.org`

[WiE06] *Wikipedia*, sito web. `http://en.wikipedia.org`

[Win03] Winskel J., *Lecture Notes on Denotational Semantics*, Cambridge University, 2003.

[1] Il lavoro risale al 1969.

Indice analitico

A^*, 26, 32
    unico a meno di omomorfismi, 33
Γ-ambiente, 96
Γ-sostituzione, 106
$\leq_{ctx}$
    estensionalità, 111
$\leq$
    proprietà, 16
$\lhd$, 106
    proprietà, 106

Alfabeto, 26
Algebra
    dei linguaggi, 25
    di Boole, 23
    per $\mathcal{P}(X)$, 23
Algoritmo, viii
Ambiente di tipi, 84
Approssimazione formale di relazioni, 105
Assiomatizzazione dei naturali, 11
Astrazione completa, 113
    fallimento, 114

Bourbaki, Nicolas, vii
Buon ordinamento, 15
    ammissibilità IP, 16
    esempio, 16

Campano da Novara, 11
Catena, 20
    Confine superiore, 20
    minimo confine superiore (lub), 20
Composizionalità, 100
Composizione, continuità, 98
Contesto, 92
CPO
    ammissibile, 69
    chiuso per catene, 69
    prodotto, 46
    somma, 49
Curry, H.B., 54

Denotazione
    termini chiusi, 97
Dominio, 39

Equivalenza contestuale, 92
Estensionalità, 110

Funzione
    continua, 37
    currying *cur*, 53
    esatta, 37
    evaluation *ev*, 52
    monotona, 19
    omomorfismo (CPO), 38
Funzioni
    continue (CPO), 41
    monotone (CPO), 41
Funzioni su $\mathbb{N}$, 14

Gordon D. Plotkin, 81

Hoare, C.A.R., 6

Insieme
    derivata di un, 27
    di generatori, 30
        minimo, 30
    iterazione, 63

parzialmente ordinato continuo (CPO), 37
parzialmente ordinato, 18
prodotto, 45

Kleene, S. C., 64

Lemma
approssimazione formale, 107
diagonalizzazione delle catene doppie, 43
intreccio di catene, 38
monotonia applicazioni continue, 38
Linguaggio, 26
Logica
di Hoare, 5
Lunghezza di una parola, 34

Massimo, 19
Massimo comune minorante, 19
Minimo, 19
Minimo comune maggiorante, 19
Monoide
definizione, 29
libero, 30
commutatività, 34
omomorfismo, 30
Monoide ordinato
definizione, 29

Numero, viii

Ordinamento
lessicografico, 17
prodotto, 18, 46
totale, 15
orelse, 113

PCF, 81
adeguatezza, 91, 105
correttezza, 91, 102
denotazione
di termini, 95
di tipi, 95
equivalenza contestuale, 90, 92
espressioni, 82
funzioni parziali ricorsive, 86
minimizzazione, 86
regole tipi, 84
sostituzione, 84
termini α-equivalenti, 82
tipi, 82
valutazione
deterministica, 88
valutazioni
forme canoniche, 87
valori, 87
variabili libere, 83
Peano, Giuseppe, 17
por, 113
indefinibilità, 114
Poset
intersezione, 47
prodotto, 46
unione, 49
Preordine
contestuale, 110
contestuale da approssimazioni formali, 110
Principio di induzione, 13
di Scott, 73
formulazione equivalente, 13
forte, 16
multipla, 17
strutturale, 17
Prodotti dipendenti, 44
Prodotto
binario tra CPO, 40
di Frobenius, 26
proprietà, 27
Proiezioni e accoppiamento, 41
Punto fisso, 54
continuità, 55

di Kleene, 57
di Tarski, 56
proprietà di $[\![\texttt{while}\ B\ \texttt{do}\ C]\!]$, 70
Punto prefisso, 54
minimo, 54

Relazione
di transizione, 89
Reticolo, 21
complementato, 22
completo, 22
distributivo, 21
non completo, 22
proprietà di $\cap, \cup$, 21

Scott, Dana S., 71
Semantica, 3
assiomatica, 5
denotazionale, 6
di tipi, 95
dinamica, 4
operazionale, 4
statica, 4
vantaggi, 3
Semantica denotazionale
dei termini PCF, 96
Serie naturale dei numeri, 11
Somma, 48
definizione induttiva, 14
esempio, viii
monotonia, 17
proprietà, 15
Stringa vuota, 26
Successore
esempio, 14

Teorema
deduttivo, 27
di Stone, 23
equivalenza termini PCF, 93
minimo punto fisso di una grammatica, 66
punto fisso di Tarski, 56
Termine
aperto, 83
chiuso, 83

Unione, 48

Von Neumann, John, 7