Alma Mater Studiorum · Università di Bologna

FACOLTÀ DI SCIENZE MATEMATICHE, FISICHE E NATURALI
Corso di Laurea Magistrale in Informatica

Ottimalità dell'ottimalità

Complessità della riduzione su grafi di condivisione

Tesi di Laurea in
Tipi e Linguaggi di programmazione

Relatore:
Chiar.mo Prof.
Simone Martini
Correlatore:
Chiar.mo Prof.
Stefano Guerrini
LIPN, Université Paris 13

Presentata da:
Marco Solieri

Sessione II
Anno Accademico 2010/2011

Ad Andrea Rappini,

a chi fu,
a chi sarebbe potuto essere.

Indice

Elenco delle figure

1. Introduzione

Trent'anni or sono il concetto di ottimalità venne formulato in senso teorico da Lévy, ma solo un decennio dopo Lamping riesce a darne elegante implementazione algoritmica. Realizza un sistema di riduzione su grafi che si scoprirà poi avere interessanti analogie con la logica lineare presentata nello stesso periodo da Girard.

Ma l'ottimalità è davvero ottimale? In altre parole, l'implementazione ottimale del λ calcolo realizzata attraverso i grafi di condivisione, è davvero la migliore strategia di riduzione, in termini di complessità?

Dopo anni di infondati dubbi e di immeritato oblìo, alla conferenza LICS del 2007, Baillot, Coppola e Dal Lago, danno una prima risposta positiva, seppur parziale. Considerano infatti il caso particolare delle logiche affini elementare e leggera, che possiedono interessanti proprietà a livello di complessità intrinseca e semplificano l'arduo problema.

La prima parte di questa tesi presenta, in sintesi, la teoria dell'ottimalità e la sua implementazione condivisa.

La seconda parte affronta il tema della sua complessità, a cominciare da una panoramica dei più importanti risultati ad essa legati. La successiva introduzione alle logiche affini, e alle relative caratteristiche, costituisce la necessaria premessa ai due capitoli successivi, che presentano una dimostrazione alternativa ed originale degli ultimi risultati basati appunto su EAL e LAL. Nel primo dei due capitoli viene definito un sistema intermedio fra le reti di prova delle logiche e la riduzione dei grafi, nel secondo sono dimostrate correttezza ed ottimalità dell'implementazione condivisa per mezzo di una simulazione.

Lungo la trattazione sono offerti alcuni spunti di riflessione sulla dinamica interna della β riduzione riduzione e sui suoi legami con le reti di prova della logica lineare.

Il lavoro necessario a questa Tesi è stato parzialmente svolto durante un periodo di tirocinio al *Laboratoire d'Informatique de Paris-Nord* (LIPN), presso l'*Université Paris 13*, sotto la supervisione del prof. Stefano Guerrini.

Parte I.

Implementazione ottimale

2. Ottimalità di Lévy

Efficiency is intelligent laziness.

David Dunham

Una strategia di riduzione per il λ calcolo è definita ottimale se, per ogni λ termine, essa opera una riduzione ottimale, cioè di lunghezza minima. Ma un classico risultato, presentato da Barendregt (1984), dimostra che una siffatta strategia non è calcolabile.

Lévy (1978) ha quindi ideato una nozione di ottimalità più debole, legata alla condivisione fra "copie" di uno stesso redex e che opera su di esse una forma di riduzione parallela. In questo breve capitolo saranno quindi presentati i principali risultati relativi a quella nozione di ottimalità per il λ calcolo che Lévy si limitò a teorizzare.

La formulazione originale mancava, infatti, di un'implementazione algoritmica, ma dimostrava ciononostante la ricorsività di una strategia che soddisfacesse questa forma di ottimalità. Per più di un decennio si è dovuto attendere un'implementazione, infine individuata nei grafi di condivisione, che saranno presentati nel successivo capitolo 3.

2.1. Equivalenza di permutazione

In questa sezione introdurremo quelle definizioni per la riduzione del λ calcolo che appartengono alla sua classica letteratura, ma che risultano necessarie per la formulazione del concetto di ottimalità. Vista l'impraticabilità dell'ottimalità *tout-court*, nel resto di questa Tesi, si farà questo nome farà semplicisticamente riferimento alla riduzione ottimale secondo Lévy.

Assumiamo di disporre di un sistema di marcatura dei redex, che sottolinei i due λ termini di cui è composto e che assegni un nome a tale marcatura.

Definizione 2.1 (Residui di un redex)**.** Sia $\mathcal{S}$ una riduzione $T \longrightarrow T'$ e sia $R \in T$ l'unico redex marcato. L'insieme R/s di redex marcati contenuti in T' è detto l'*insieme dei residui* di R relativo ad $\mathcal{S}$. Inoltre, il redex R è detto *antenato* di ogni redex appartenente ai suoi residui rispetto ad $\mathcal{S}$

La più semplice forma di composizione di riduzioni è quella sequenziale. Presi due redex R_1 ed R_2, la loro composizione consiste nella riduzione ordinata dei due, ovvero prima R_1, poi R_2; ed è semplicemente indicata con $R_1 R_2$.

Ora possiamo introdurre una composizione di riduzioni rispetto ai residui, indicata con l'operatore $\sqcup$. La riduzione $\mathcal{R}_1 \sqcup \mathcal{R}_2$ è definita come $\mathcal{R}_1(\mathcal{R}_2/\mathcal{R}_1)$, cioè la composizione sequenziale composta prima da $\mathcal{R}_1$, poi dalla riduzione dei residui di $\mathcal{R}_2$ rispetto ad essa.

Questa forma di composizione consente di stabilire il seguente lemma, che useremo per la successiva definizione di una relazione fra riduzioni che sono rispettive permutazioni.

Lemma 2.1 (Mosse parallele)**.** *Siano $\mathcal{R}_1$ e $\mathcal{R}_2$ due insiemi di redex appartenenti ad un λ termine T. Allora:*

- *$\mathcal{R}_1 \sqcup \mathcal{R}_2$ e $\mathcal{R}_2 \sqcup \mathcal{R}_1$ terminano nella stessa espressione.*
- *Per ogni $\mathcal{R}_3$, insieme di redex, vale che $\mathcal{R}_3/(\mathcal{R}_1 \sqcup \mathcal{R}_2) = \mathcal{R}_3/(\mathcal{R}_2 \sqcup \mathcal{R}_1)$*

Definizione 2.2 (Equivalenza di permutazione)**.** L'*equivalenza di permutazione* $\approx$ è la minima congruenza rispetto alla composizione che soddisfi il lemma delle mosse parallele e l'eliminazione di passi vuoti.

Semplificando, sono messi in relazione tutte quelle sequenze di redex per i quali il loro ordine di applicazione è irrilevante, dacché operano in maniera parallela. Visto che l'obiettivo della riduzione condivisa è proprio quello di ridurre insieme tutte le copie di un certo redex, questa proprietà di permutabilità è proprio quella ci aspettiamo sia soddisfatta dalla riduzione.

2.2. Famiglie di redex

L'intuizione di raggruppare tutti quei redex che sono creati allo stesso modo è formalmente definita dal concetto di famiglia di redex, che vorremmo poter ridurre tutta insieme.

Per poterci riferire al "passato" di un redex, indicheremo con $\mathcal{S}R$ quel redex R che ha come storia l'insieme di redex $\mathcal{S}$, ovvero che è ottenibile per mezzo della riduzione che corrisponde alla sua storia. Si noti la coincidenza di notazione rispetto a quella della composizione di redex.

Definizione 2.3 (Relazione di copia)**.** Un redex S con storia $\mathcal{S}$ è detto *copia* di un redex R con storia $\mathcal{R}$, indicato con $\mathcal{R}R \precsim \mathcal{S}S$, se e solo se esiste una derivazione $\mathcal{T}$ per la quale $\mathcal{R}\mathcal{T} \approx \mathcal{S}$.

Definizione 2.4 (Equivalenza di famiglia)**.** La *relazione di famiglia* $\simeq$ è la chiusura simmetrica e transitiva della relazione di copia.

2.3. Riduzione per famiglie

Ora è possibile definire la riduzione per famiglie come quella che, scelta una famiglia, ad ogni passo riduce tutti i redex che sono relazione di famiglia per essa. L'idea insomma è che, in tal modo, il "lavoro" necessario alla riduzione di una famiglia venga effettuato una volta per tutte. Una derivazione detta completa, quindi, contrae un insieme massimale di copie di un unico redex.

Definizione 2.5 (Completezza di una riduzione)**.** Una riduzione non nulla $\mathcal{R}_1 \dots \mathcal{R}_i \dots \mathcal{R}_n$ è completa se e solo se $\mathcal{R}_i$ è l'insieme massimale di redex per cui, per ogni coppia di redex S ed T in $\mathcal{R}_i$,

$$\mathcal{R}_1 \dots \mathcal{R}_{i-1} S \simeq \mathcal{R}_1 \dots \mathcal{R}_{i-1} T$$

La completezza assicura pertanto che la riduzione non esegua riduzioni duplicate, dato che esse sono necessariamente eseguite insieme in ogni passo di riduzione che sia completa.

2.4. Riduzione ottimale

Ma la completezza di riduzione non è una proprietà sufficiente a garantire l'ottimalità secondo Lévy: è necessario anche che la riduzione non esegua lavoro inutile, che la strategia di riduzione sia quindi pigra. Per formalizzare questa seconda condizione, introduciamo il concetto di chiamata per necessità.

Presa una riduzione $\mathcal{R}$, definiamo $\mathsf{Res}(\mathcal{R})$, come l'insieme di redex che possiedono un residuo contratto in $\mathcal{R}$. Ricordando, inoltre, che una derivazione è detta terminante quando la sua espressione finale è in forma normale, possiamo definire la strategia di riduzione con chiamata per necessità.

Definizione 2.6 (Necessità di un redex)**.** Un redex R appartenente ad un λ termine T è *necessario* se e solo se, per ogni riduzione terminante S a partire da T, $R \in \mathsf{Res}(S)$.

Definizione 2.7 (Riduzione con chiamata per necessità)**.** Una riduzione $\mathcal{R} = R_1 \dots R_n$ è detta a *chiamata per necessità* se e solo se esiste almeno un redex necessario in ogni $R_i \in \mathcal{R}$.

In altre parole un redex è necessario se esso deve essere ridotto in ogni riduzione terminante che normalizza il termine iniziale. È facile notare che, preso un λ termine, il suo redex sinistro-esterno[1] è sicuramente necessario, quindi la strategia sinistra-esterna, o normale, è una strategia con chiamata per necessità.

Stabiliamo ora informalmente come misura del costo della riduzione, quella che considera come unitaria la riduzione di una famiglia di redex, ovvero che riduce in

[1] Gli anglofoni preferirebbero *leftmost-outermost*.

un unico passo tutte le copie di uno stesso redex. Rispetto a tale misura, è possibile dimostrare che il costo di una riduzione per famiglie limita inferiormente ogni altra riduzione terminante.

Teorema 2.1 (Ottimalità (informale))**.** *Ogni riduzione parallela completa con chiamata per necessità raggiunge la forma normale in costo ottimale.*

L'intelligenza della riduzione per famiglie e la pigrizia della riduzione con chiamata per necessità costituiscono pertanto gli ingredienti dell'efficienza della riduzione ottimale.

3. Grafi di condivisione

> When I'm working on a problem, I never think about beauty. I think only how to solve the problem. But when I have finished, if the solution is not beautiful, I know it is wrong.
>
> R. Buckminster Fuller (1895 - 1983)

La riduzione condivisa è una recente tecnica di riduzione su grafi per espressioni funzionali che soddisfa l'ottimalità secondo Lévy presentata nel precedente capitolo 2. Questo sistema di riscrittura su grafi venne inizialmente proposto da Lamping (1989), alla conferenza internazionale ACM POPL (*Principles of Programming Languages*) e successivamente semplificato da Gonthier *ed altri* (1992), che esplorarono le analogie con la geometria d'interazione di cui si accennerà in 4.1.

Tecniche di riduzione su grafi che realizzassero una forma di condivisione non erano certo sconosciute al tempo, ma risultano, in confronto, primitive. Ad esempio, i grafi presentati da Wadsworth (1971), che furono uno dei primi risultati presenti in letteratura, sono in grado di realizzare condivisione di sottotermini. La soluzione di Lamping, invece, è molto più potente, dato che consente condivisioni di granularità minima ed è in grado di operare, di conseguenza, anche su contesti di riduzione.

In questo capitolo sarà prima presentato il sistema di rappresentazione e riscrittura dei grafi di condivisione, secondo lo stile di Asperti e Guerrini (1998), che tradizionalmente suddivide l'algoritmo astratto, che ne contiene il cuore, da quello completo, che realizza l'oracolo ad esso necessario. La parte finale del capitolo presenta i due risultati più importanti: la correttezza rispetto al λ calcolo e l'ottimalità secondo Lévy.

3.1. Algoritmo astratto

3.1.1. Sintassi

La sintassi dei grafi di condivisione deriva dagli alberi di sintassi astratta del λ calcolo. Questi possono essere risparmiati dalla presenza di variabili esplicite, introducendo rami aggiuntivi che colleghino ogni astrazione alle sue legature. In questo

modo, i nodi dell'albero o, meglio, del grafo diretto aciclico, corrispondono precisamente alle due regole grammaticali per la formazione di λ termini: l'astrazione, indicata con λ, e l'applicazione, indicata con @.

Il nodo d'astrazione è rappresentato a sinistra in figura 3.1. Esso ha due archi discendenti detti secondari, uno destro, connesso al corpo; uno sinistro, eventualmente ramificato, connesso ad ogni occorrenza delle variabili che lega. Il suo arco ascendente, detto principale, è quello funzionale, dato che da un punto di vista di teoria dei tipi, se la variabile ha tipo A e il corpo tipo B, l'astrazione ha tipo $A \to B$.

Anche l'applicazione ha due archi discendenti: uno è connesso ad un albero funzionale ed è principale, l'altro porta invece al suo argomento, ed è secondario. Il collegamento superiore del nodo @, rappresentato in figura 3.1, è quello che chiameremo del risultato ed è un ramo secondario.

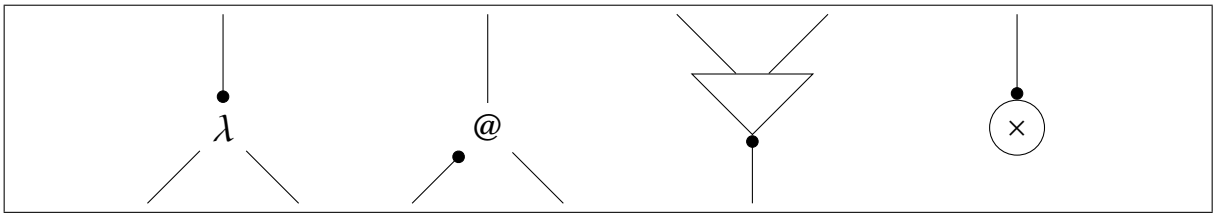


Figura 3.1.: Nodi dei grafi di condivisione: astrazione, applicazione, fan, spazzatura

La parte relativa ai nodi di astrazione e applicazione della sintassi dei grafi di condivisione non apporta alcun cambiamento al rispetto alla rappresentazione con alberi sintattici, eccezion fatta per la distinzione dei rami lungo i quali i nodi possono interagire, quelli detti *principali*.

D'altro canto, si aggiunge un nodo esplicito, introdotto al posto della ramificazione presente nei rami di legatura dei nodi λ. Il nodo, chiamato *fan*, è anch'esso dotato di un ramo principale e di due rami secondari ed è disegnato a destra nella figura 3.1. Da un punto di vista statico, il fan rappresenta la condivisione: il ramo principale è connesso ad un termine che risulta condiviso fra i termini collegati ai rami secondari dello stesso fan.

Un'ulteriore nodo è necessario per poter correttamente gestire la presenza di variabili pendenti e rendere il grafo privo di rami discendenti liberi. Applicando un'astrazione con variabile non legata ad un termine, il risultato è la cancellazione di questo termine, dato che esso non viene sostituito al suo interno. È per questo motivo che il nodo è chiamato *spazzatura* e che la sua riduzione, come vedremo, realizza l'elminazione delle porzioni disconnesse ed inutili del grafo di condivisione.

Possiamo, dunque, completare una prima e parziale formulazione dei grafi di condivisione astratti, attraverso una traduzione iniziale a partire da termini del λ calcolo.

Definizione 3.1 (Traduzione iniziale (idea))**.** Sia T un λ termine. La funzione di traduzione iniziale $\mathcal{T}$ genera l'albero di sintassi astratta di T e restituisce il grafo G ottenuto sostituendo ogni diramazione presente in T con un nodo fan.

3.1.2. Riduzione

Chiarita una prima idea della sintassi dei grafi di condivisione, che ne rappresenta la dimensione statica, possiamo ora procedere introducendone la dinamica, che realizza la riduzione sui grafi. Tale riduzione è formalizzata per mezzo di regole di riscrittura locali su grafi: la β riduzione ottimale e la propagazione dei fan.

Il *passo* β, rappresentato in figura 3.2, coinvolge un nodo @ e un nodo λ, connessi attraverso il loro ramo principale. Questi due nodi vengono eliminati, il corpo dell'astrazione sale verso l'alto, diventando così il nuovo estremo superiore del grafo, e l'argomento dell'applicazione è collegato alla porta di legame del primo. Si noti che, rispetto alla β riduzione classica, non è eseguita alcuna sostituzione, grazie al fatto che quest'ultimo collegamento realizza la condivisione dell'argomento della riduzione con tutte le occorrenze della variabili che erano legate dall'astrazione.

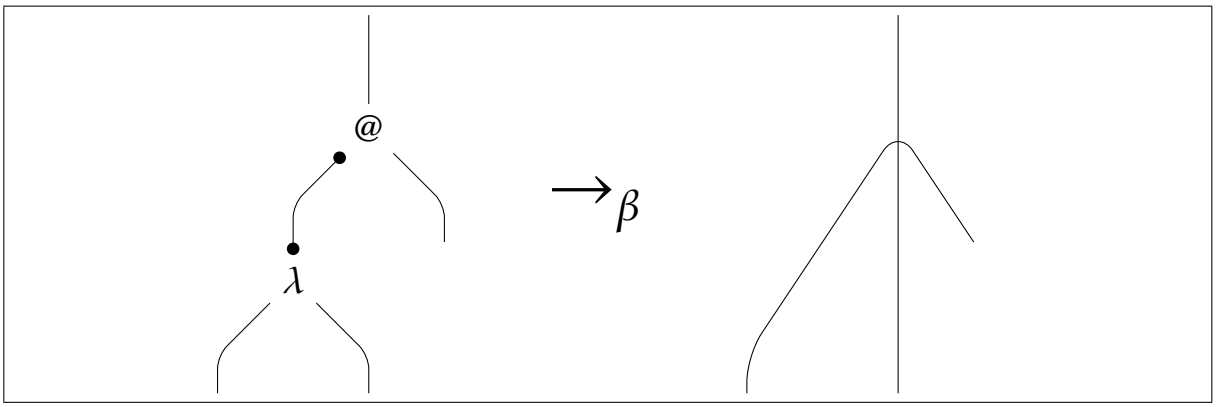
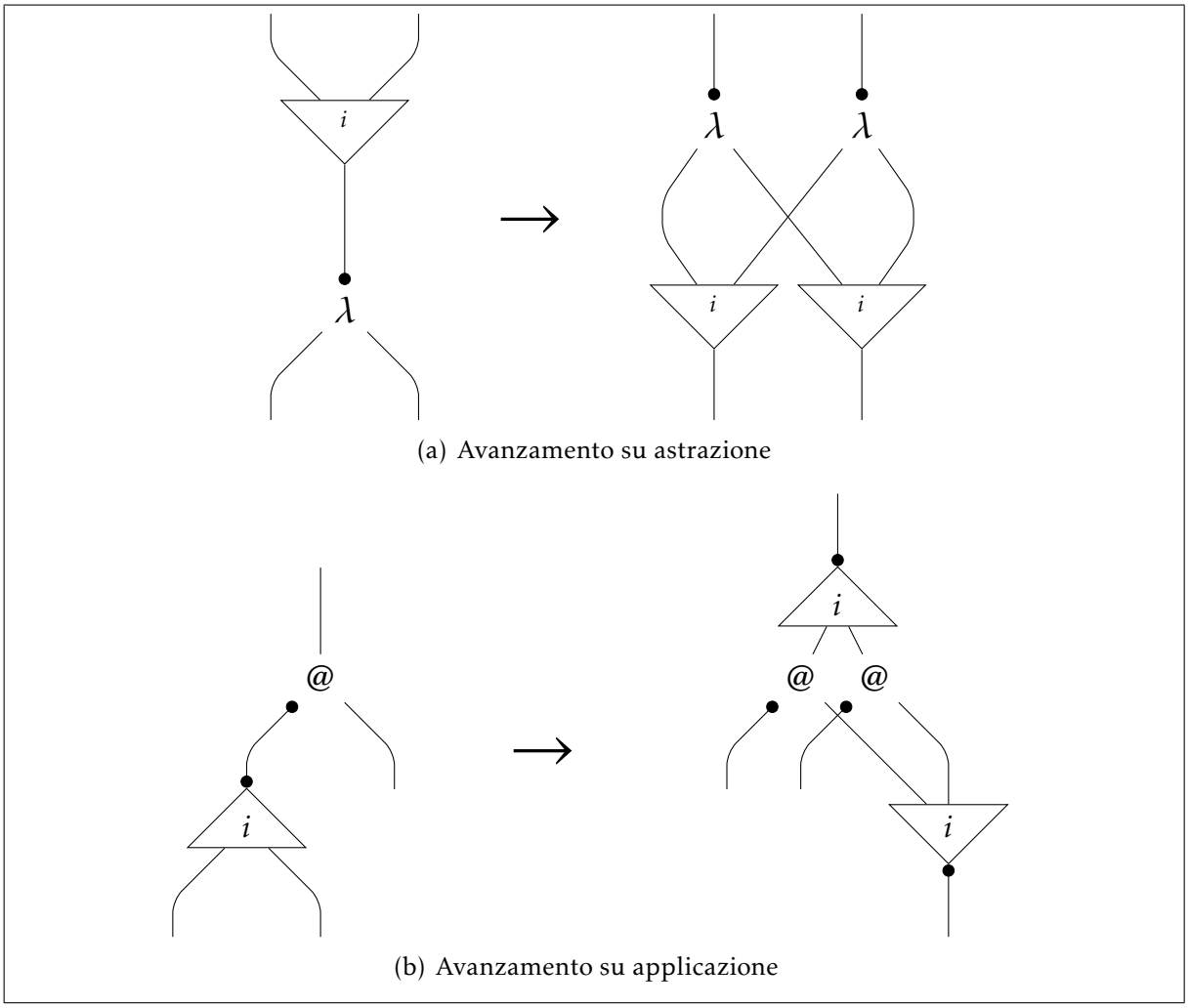


Figura 3.2.: Riduzione condivisa: β riduzione

La *propagazione di fan* realizza la duplicazione di nodi λ e @, come mostrato in figura 3.3. Il nodo fan, quando fronteggia il ramo principale di un nodo di astrazione o di applicazione, può quindi avanzare, generandone due copie e situandosi sui due rami secondari di ognuna delle copie.

Immaginando l'iterazione della regola di propagazione, si comprende come il fan sia in grado, passo dopo passo, di duplicare grafi di dimensione arbitraria. Ciononostante, il sistema di riduzione non impone urgenza alla regola di propagazione dei fan, ed è per questo motivo che diventa possibile l'implementazione di duplicazione parziale.

Si può già comprendere che, in un grafo, quanto maggiori sono i redex condivisi, tanto inferiori saranno i costi della riduzione, dato che ogni riduzione su un redex condiviso è effettuato una sola volta per ognuno dei contesti nei quali è usato. In assenza di condivisione, ognuno di tali contesti richiederebbe una copia distinta di tale redex ed ognuno di essi andrebbe separatamente ridotto. Più avanti, nella sezione 3.4, vedremo, infatti, come la strategia preferibile sia quella che propaga i fan in maniera pigra.



(a) Avanzamento su astrazione

(b) Avanzamento su applicazione

Figura 3.3.: Riduzione condivisa: avanzamento di fan

La porzione più interessante del sistema di riduzione astratta gestisce il comportamento degli *incontri fra fan*, per i quali distinguiamo due casi differenti. A fronteggiarsi potrebbero essere due fan che, intuitivamente, si stanno occupando di effettuare la stessa duplicazione, nel qual caso i due fan dovrebbero effettuare un'*annichilamento*, unendo i rispettivi collegamenti. Ma potrebbero essere anche essere due fan che non hanno alcuna correlazione e, in questo caso, dovrebbero invece fare uno *scambio*, duplicandosi vicendevolmente. Per distinguere i due casi, possiamo supporre di assegnare ad ogni fan, durante la traduzione iniziale, un indice univoco che sia preservato lungo i relativi passi di propagazione.

Definizione 3.2 (Traduzione iniziale (astratta))**.** Sia T un λ termine. La funzione di traduzione iniziale $\mathcal{T}$ genera l'albero di sintassi astratta di T e restituisce il grafo G ottenuto sostituendo ogni diramazione presente in T con un nodo fan, etichettato con un un'etichetta univoca.

Questa modifica spiega la presenza dell'etichetta i all'interno delle raffigurazioni dei fan. Per mezzo di questo sistema di etichettatura, è ora possibile la formulazione, con una semplice condizione a margine, della regola che gestisce l'incontro di fan, mostrato in figura 3.4.

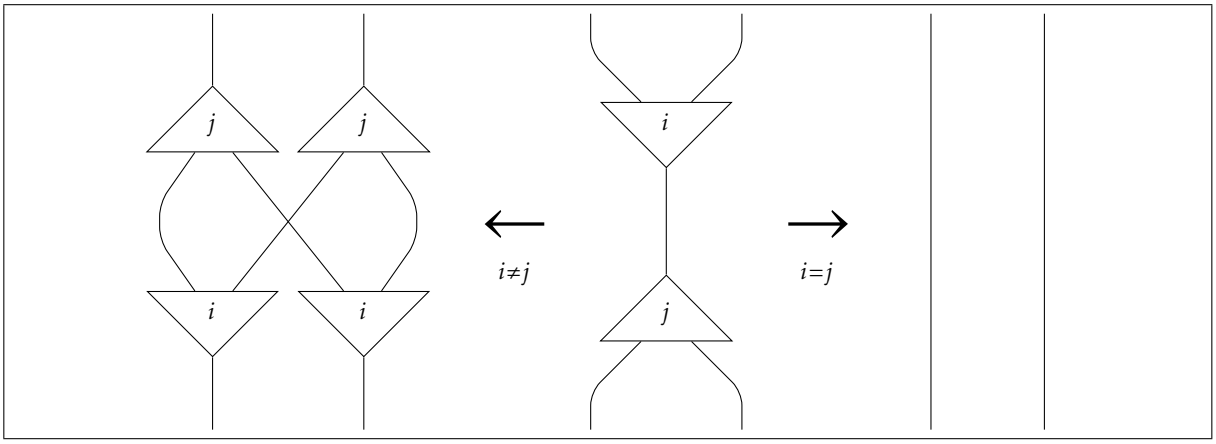


Figura 3.4.: Riduzione condivisa: incontro fra fan, scambio o annichilamento

Infine, il raccoglimento della spazzatura è eseguito per mezzo della propagazione del relativo nodo, che ha il compito di rimuovere dal grafo ogni area disconnessa in esso contenuta per mezzo della sua normalizzazione. Esso "mangia" tutto ciò che incontra lungo collegamenti principali, ma la il suo incontro un fan è una situazione delicata da gestire.

Per semplicità di trattazione le sue regole di riduzione sono qui tralasciate, rimandando alle già citate fonti per una trattazione anche di questo caso.

Le definizioni introdotte fin qui costituiscono il sistema di riscrittura che realizza l'algoritmo astratto.

Definizione 3.3 (Riduzione condivisa astratta)**.** Il sistema di riduzione condivisa astratta è costituito dalle regole così denominate:

- β riduzione, in fig. 3.2;
- propagazione di fan:
  - duplicazione o avanzamento di fan:
    - attraverso astrazione, in fig. 3.3(a),
    - attraverso applicazione, in fig. 3.3(b);
  - incontro di fan, in fig. 3.4.

3.2. Oracolo

L'algoritmo astratto appena presentato non è sufficiente a governare correttamente ogni situazione (si veda il successivo esempio), perché il comportamento di ordine superiore del λ calcolo è in grado di confondere il sistema di etichettatura statica dei fan. È quindi necessaria, in generale, una corretta implementazione di un oracolo che, nel caso di un incontro tra fan, distingua correttamente quale delle due azioni debba essere intrapresa, se di scambio o di annichilamento.

L'idea di base, che permette una corretta implementazione dell'oracolo, è pertanto che le etichette univoche dei fan siano sostituite da indici interi che debbano, in certi casi, essere cambiati.

Si tenga presenta che questa problematica è estranea ad ogni calcolo di tipo stratificato, nei quali, cioé, questi cambiamenti di etichetta non sono affatto possibili. Questa intuizione ancor vaga sarà chiarita dal significato dell'etichettatura e, soprattutto, dalla presentazione dei sistemi stratificati del successivo capitolo 5.

3.2.1. Sintassi

Assegniamo un indice intero non solo ai fan, ma ad ogni nodo dei grafi di condivisione che rappresenta, a livello intuitivo, il suo livello di profondità, ovvero il numero di modi diversi con i quali esso può essere condiviso all'intero del grafo. Pertanto la traduzione iniziale assegnerà il livello 0 alla radice dell'albero di sintassi astratta ed aumenterà il livello ogni volta che un sottoalbero risulta argomento di un'applicazione.

Si introducono, inoltre, due nuovi nodi binari che gestiranno la dinamica dei livelli con la loro propagazione. Il *croissant* chiude un livello ed è rappresentato a sinistra in figura 3.5, il *bracket* apre un livello ed è a destra nella stessa figura.

Concettualmente, quindi il bracket e il croissant, nella traduzione iniziale, sono posizionati ai bordi dei livelli, delimitando le aree che potrebbero essere condivise.

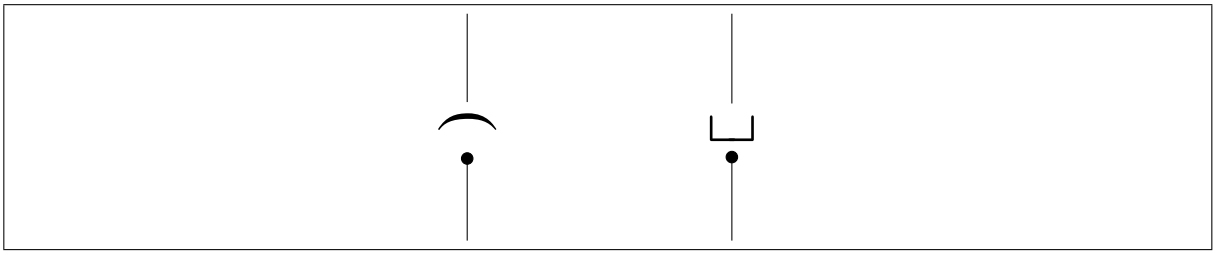


Figura 3.5.: Nodi dei grafi di condivisione: croissant e bracket

Definizione 3.4 (Traduzione iniziale (completa))**.** Sia T un λ termine. La funzione di traduzione iniziale $\mathcal{T}$ genera l'albero di sintassi astratta di T e restituisce il grafo G ottenuto inserendo, induttivamente, nell'ordine:

- un nodo croissant per ogni variabile,
- un nodo bracket per ogni variabile libera ogni volta che essa appartiene ad un termine che è argomento di un'applicazione,
- un nodo fan per ogni coppia di variabili con legatura comune.

Inoltre i livelli sono assegnati, induttivamente come segue:

- la radice del grafo ha livello 0,
- se un'applicazione è di livello i, il suo argomento ha livello $i + 1$.

Questa formulazione dei grafi di condivisione consente, fra l'altro, di apprezzare fino in fondo l'analogia fra i livelli e le scatole della logica lineare, delle quali si parlerà in 5.2.3.

3.2.2. Riduzione

Le regole di riduzione per i nodi dell'oracolo sono raffigurate nelle figure 3.6 e 3.7. Se fronteggiano porte principali di un nodo a livello superiore, idealmente più interno, si propagano attraverso di esso, cambiandone l'indice di profondità: il croissant lo diminuisce di un'unità, il bracket lo aumenta. Quando invece incontrano un nodo dello stesso tipo e dello stesso livello, si annullano con esso.

Così esteso il sistema di riduzione per l'oracolo è completo ed è possibile aggiungerlo alla definizione del sistema affinché realizzi l'algoritmo completo.

Definizione 3.5 (Riduzione condivisa completa)**.** Il sistema di riduzione condivisa completa è costituito dalle regole così denominate:

- β riduzione, in fig. 3.2;
- propagazione di fan:

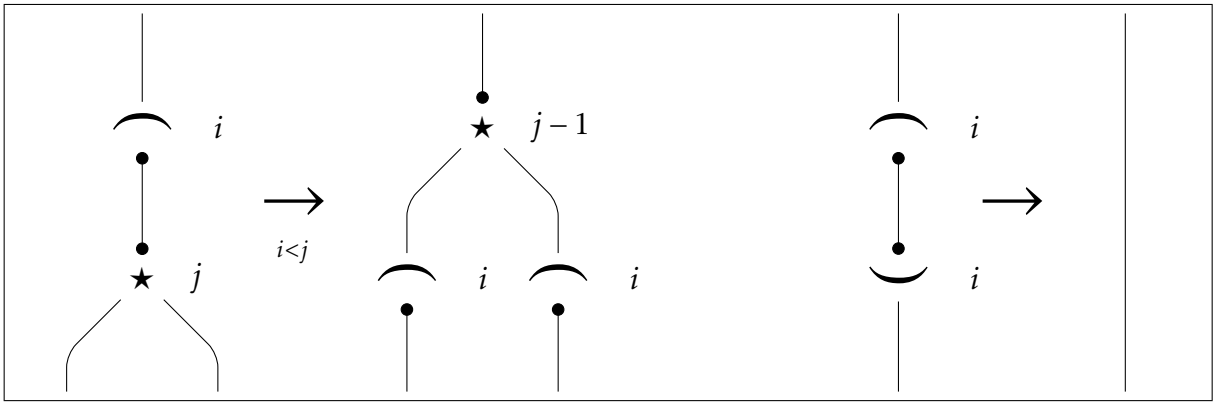


Figura 3.6.: Riduzione condivisa: propagazione di croissant: avanzamento (generico) e annichilamento

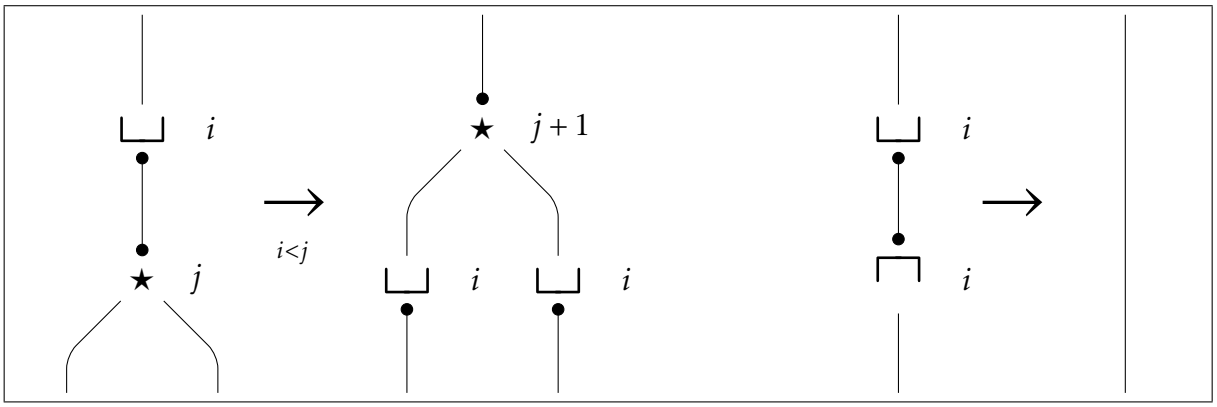


Figura 3.7.: Riduzione condivisa: propagazione di bracket: avanzamento (generico) e annichilamento

  - duplicazione o avanzamento di fan, in fig. 3.3,
  - incontro di fan, in fig. 3.4;

- gestione dei livelli:
  - propagazione di croissant, in fig. 3.6,
  - propagazione di bracket, in fig. 3.7.

3.3. Correttezza

Intuitivamente, un grafo di condivisione *corrisponde* ad un albero sintattico se esiste una "appropriata" biezione che preserva i percorsi in essi contenuti. Attraverso questa relazione, qui non riportata per semplicità di trattazione, è possibile mostrare che l'implementazione condivisa, costituita dai grafi e dalla relativa riscrittura appena introdotta, è corretta.

Teorema 3.1 (Correttezza dell'implementazione condivisa del λ calcolo)**.** *Sia T un λ termine e $G = \mathcal{T}(T)$ la sua traduzione iniziale. Se $G \longrightarrow^* G'$, allora esiste una riduzione $T \longrightarrow^* T'$ tale che G' corrisponde a T'.*

La dimostrazione richiede una complessa analisi della semantica dei contesti, che descrive i percorsi contenuti all'interno dei grafi di condivisione e ne "rilegge" il contenuto sequanzializzandolo. Per una sua trattazione esaustiva si rimanda, fra gli altri, al testo di riferimento di Asperti e Guerrini (1998).

Il teorema di correttezza mostra, in primo luogo, che la riduzione su grafi di condivisione è effettivamente equivalente alla β riduzione del λ calcolo, per la quale intende appunto essere una implementazione coerente. Da questo punto di vista, insomma, il risultato si limita a confermare l'adeguatezza del sistema di riscrittura.

Dal punto di vista della complessità, possiamo però cogliere un significato più profondo, notando che la riduzione condivisa consiste, com'è evidente dalla sua definizione, di regole di riscrittura locali e, quindi, unitarie, mentre il passo di β riduzione non lo sia affatto. Per questo motivo si intuisce che la riduzione condivisa rappresenti una rappresentazione più fine della β riduzione, che ne esplicita due dinamiche in essa altrimenti celate: la duplicazione e la gestione dei contesti. Della complessità della riduzione ottimale e dei suoi legami con la complessità del λ calcolo si tratterà fra poco, in §4.3.

3.4. Ottimalità

La condivisione realizzata dall'algoritmo realizza pienamente la riduzione completa per famiglie di Lévy, precedentemente definita in 2.3.

Teorema 3.2 (Completezza di famiglie)**.** *Sia T un λ termine e $G = \mathcal{T}(T)$ la sua traduzione iniziale. Allora esiste una riduzione condivisa $G \longrightarrow^* G'$ se e solo se esiste una riduzione completa di una famiglia $T \longrightarrow^* T'$ tale che G' corrisponde a T'.*

Per raggiungere l'ottimalità è quindi sufficiente imporre che la strategia di riduzione utilizzata per scegliere quale riscrittura eseguire, sia con chiamata per necessità (cfr. §2.4), il cui più semplice esempio è la strategia normale, chiamata anche sinistra-esterna.

Teorema 3.3 (Ottimalità)**.** *Il sistema di riduzione con grafi di condivisione dotato di una strategia di riduzione con chiamata per necessità è ottimale.*

Parte II.

Complessità

4. Stato dell'arte

Nisi credideritis, non intellegetis.

Aurelius Augustinus Hipponensis
(354 - 430)

Dopo la prima parte della trattazione, introduttiva all'ottimalità e alla sua implementazione, questo capitolo ne apre la parte centrale presentando una rassegna dei risultati più rilevanti. A partire dalla loro analisi, saranno così esplicitati gli intenti che hanno indirizzato il lavoro raccolto in questa tesi.

4.1. Logiche lineari

Le reti di prova della logica lineare e la loro riduzione hanno il merito di meglio dettagliare la natura della β riduzione: l'operazione di sostituzione va a legare le variabili legate di una astrazione alla radice del termine cui è applicata, l'operazione di duplicazione replica questo termine per ognuna delle occorrenze. Nonostante questo spunto, analogo a quello offerto dal λ calcolo a sostituzione esplicita, la loro similarità con il calcolo a sequenti appare eccessivamente forzata, quando si considerano frammenti esponenziali, ad esempio con codifica fra $!(A \multimap B)$ e $A \to B$. In questi casi, infatti, essa realizza la duplicazione per mezzo di riduzione non locale e non unitaria che coinvolge interi blocchi esponenziali della rete, le scatole.

La riduzione condivisa, al contrario, è un sistema che implementa la duplicazione proprio in maniera locale ed unitaria, che è stato proposto indipendentemente e contemporaneamente alla riduzione su reti di prova. La relazione a livello di significato fra le due è profonda sia nelle contrazione o condivisione, staticamente, sia nella duplicazione, dinamicamente. Per questo tale tema fu esplorato da Gonthier *ed altri* (1992) e poi analizzato a fondo da Guerrini (1999), che sviluppa una teoria generale dei grafi di condivisione. Il riferimento più completo sull'implementazione ottimale non manca di riportare questo risultato almeno sommariamente (Asperti e Guerrini , 1998, §4).

Quei risultati, in particolare l'introduzione della tecnica di rilettura dei grafi per mezzo di regole di propagazione non ottimali e la generale impostazione algebrica, hanno permesso gli sviluppi presentati da Guerrini *ed altri* (2000, 2003). Essi presentano un'implementazione condivisa della logica MELL, sfruttando un sistema

intermedio di grande rilevanza concettuale che separa concettualmente due operazioni fino ad allora proprio confuse fra quelle dell'oracolo. Infatti si era affermata la tendenza a fare uso di nodi *mux*, che invece realizzavano sia la manutenzione dei livelli, che la duplicazione. Ma, in realtà, la prima non fa parte della "logica" della riduzione, mentre la seconda del "controllo".

4.2. Correttezza e completezza

Dato che la riduzione ottimale si configura come l'implementazione ottimale di un certo sistema di calcolo, i risultati di correttezza sono necessari e quelli di completezza auspicabili.

La correttezza per l'implementazione ottimale del λ calcolo, presentato nel teorema 3.1, viene per la prima volta dimostrata dallo stesso ideatore dei grafi di condivisione Lamping (1989). A meno delle semplificazioni di Gonthier *ed altri* (1992), la dimostrazione del risultato diventa la formulazione classica, realizzata per via semantica, attraverso gli strumenti della semantica dei contesti.

La tecnica si può infatti ritrovare, *mutatis mutandis*, nel recente (Baillot *ed altri* , 2011, §2), dove gli autori considerano una classe di traduzioni ammissibili dai termini tipabili con le logiche EAL e LAL in grafi di condivisione. L'intento della generalizzazione è quello di considerare, preso un λ termine ogni possibile rete che costituisca prova del suo tipo. I risultati di correttezza, già attesa dopo l'ipotesi di Asperti (1998), e di completezza vengono enunciati, in forma debole, riferendosi, cioè, alle sole riduzioni normalizzanti. Nello specifico, è definita una funzione di rilettura $\mathcal{R}$, che ricostruisce i λ termini a partire dalla loro rappresentazione condivisa, e che viene realizzata per mezzo della semantica dei contesti, che analizza i percorsi contenuti all'interno dei grafi di condivisione.

Teorema 4.1 (Correttezza dell'implementazione condivisa di $\Lambda_{EAL/LAL}$)**.** *Sia T un termine tipabile in EAL o LAL e $G = \mathcal{T}(T)$ una traduzione iniziale ammissibile. Se esiste $G \longrightarrow^* G'$ tale che G' è in forma normale, allora esiste una riduzione $T \longrightarrow^* T'$, per cui $\mathcal{R}(G') = T'$.*

Teorema 4.2 (Completezza dell'implementazione condivisa di $\Lambda_{EAL/LAL}$)**.** *Sia T un λ termine tipabile in EAL o LAL e $G = \mathcal{T}(T)$ una traduzione iniziale ammissibile. Se esiste $T \longrightarrow^* T'$ tale che T' è in forma normale, allora esiste una riduzione $G \longrightarrow^* G'$, per cui $\mathcal{R}(G') = T'$.*

Il caso, per certi versi più generale, della logica MELL viene preso in considerazione da Guerrini *ed altri* (2000, 2003), per la quale mostrano la correttezza ed ottimalità dell'implementazione condivisa. La correttezza è, però, formulata in forma forte, senza cioè limitarsi a considerare le sole normalizzazioni, ma generiche riduzioni.

Essa e dimostrata per via sintattica, attraverso la simulazione della riduzione condivisa per mezzo della eliminazione del taglio su reti di prova, modulo una relazione di "minor condivisione" fra reti.

Questa strategia di dimostrazione risulta simile alla tecnica che verrà sfruttata, nella presente tesi, nei capitoli 6 e 7, che presenteranno, rispettivamente, il sistema intermedio ED e la simulazione su di esso della riduzione condivisa per sistemi EAL o LAL. La citata relazione di minor condivisione non sarà, però, esplorata nella sua generalità, ma piuttosto utilizzata in una versione più dettagliata e ristretta ai soli scopi di garantire la simulazione. Grazie ad essa sarà possibile ricavare come semplice corollario (sez. 7.3) il risultato di correttezza forte per l'implementazione condivisa di termini di EAL o LAL, ancora mancante.

4.3. Complessità

La cronistoria degli sviluppi nello studio della complessità dell'implementazione ottimale potrebbe essere riassunta nella forma di un racconto, ancora privo di finale.

"Dopo la delusione, per aver creduto invano che le famiglie di redex fossero una misura del costo della riduzione sul λ calcolo, molti hanno frainteso questo fallimento, sfiduciando per intero la riduzione per famiglie. Ma c'è chi continua a sperare in un suo riscatto."

4.3.1. Delusione

I primi risultati nello studio sull'effettiva complessità della riduzione ottimale sono emersi contestualmente alla ricerca di un modello di costo per il λ calcolo. La riduzione ottimale (cap. 2), astrattamente enunciata nella tesi dottorale di Lévy (1978) era infatti apparsa come significativa per la ricerca di una misura ragionevole di costo. Si sperava che il numero di famiglie di un λ termine costituisse infatti una misura del costo intrinseco necessario alla sua normalizzazione.

Le prime avvisaglie che questo fosse un errore venne da Lawall e Mairson (1996) che dimostravano l'esistenza di λ termini di taglia n, per i quali la normalizzazione richiede $\Theta(n)$ riduzioni, fra passi β condivisi e passi di interazione di fan, ma $\Omega(2^n)$ operazioni dell'oracolo. Asperti (1996), proponeva infatti una misura di complessità del λ calcolo basata non solo sul numero di passi β paralleli, ma anche sul numero di annichilamenti fra fan di cui essi necessitano.

Insieme, Asperti e Mairson (1998), hanno definitivamente stabilito che si trattasse di un errore, chiarendo che la manutenzione degli indici, il cosiddetto *bookkeeping*, ha un costo intrinseco più che elementare.

Teorema 4.3 (Complessità della riduzione ottimale)**.** *Per ogni intero l fissato, esiste un insieme di λ termini esplicitamente tipati e chiusi tali che, per ognuno di essi, se n è la sua dimensione, allora:*

- *la normalizzazione condivisa richiede $\mathcal{O}(n)$ riduzioni per famiglie*
- *il costo necessario ad un qualsiasi implementazione su una macchina Turing-equivalente richiede tempo in $\Omega(K(n))$.*

Gli autori elaborano un sistema di manipolazione dei termini semplicemente tipati, basato sull'η-espansione, per mezzo del quale ogni termine così processato può essere ridotto in un numero di passi di riduzione per famiglia che è lineare nella sua dimensione. Il vecchio teorema di Statman (1979) sul λ calcolo tipato afferma però che ogni riduzione di termini semplicemente tipati non può essere limitato da alcuna funzione elementare.

In tal modo viene conclusa la tesi, che nega ogni possibilità d'utilizzo del numero di famiglie come modello di costo per il λ calcolo, ma che apre interessanti interrogativi sulla definizione di un'alternativa valida.

4.3.2. Fraintendimento

Il teorema 4.3 è stato interpretato come un risultato negativo non solo sull'uso delle famiglie come misura di costo, ma in generale sulla riduzione ottimale. Infatti, dal marginale interesse (per non dire oblìo) che il tema ha ricevuto negli anni successivi, si può intuire che se ne sia diffuso il travisamento che interpreta la non-elementarità come inefficienza intrinseca della riduzione con grafi di condivisione o, più in generale, di ogni riduzione Lévy-ottimale.

Un siffatto fraintendimento del risultato è difficilmente comprensibile, dato che è evidente che gli autori non concludono nulla sulla riduzione ottimale, ma solo sulla relazione fra il numero di famiglie necessario ad una riduzione e il costo effettivo. Considerando il limite inferiore ai costi intrinseci che venne stabilito da Statman, sarebbe quantomeno pretenzioso aspettarsi che l'implementazione abbassi riduca i costi di riduzione sotto il limite teoricamente stabilito. Efficiente sì, miracolosa no.

In questo contesto, Asperti *ed altri* (2004) tentarono di fare chiarezza, lavorando sulla logica affine elementare (EAL), che elimina la necessità delle operazioni di mantenimento realizzate dall'oracolo. Asperti, Coppola e Martini mostrano che anche, in questo caso, ci sono termini per i quali i soli passi di duplicazione necessari alla normalizzazione sono più che elementari nella sua dimensione. Ma la duplicazione è un costo che non viene introdotto dall'algoritmo di Lamping: é un costo che ogni implementazione deve necessariamente sostenere. In tal modo si è, moralmente, rimarcato che i costi della riduzione condivisa sono necessari ad *ogni* riduzione.

4.3.3. Speranza

Il primo ed unico risultato esplicitamente positivo è stato recentemente prodotto da Baillot *ed altri* (2011), che considerano la riduzione condivisa nel caso delle

logiche affini. Prima della già citata correttezza, il loro più importante risultato mostra, infatti, come l'implementazione condivisa di ogni traduzione ammissibile di ogni termine tipato da EAL o LAL richieda tempo adeguato alla classe di complessità implicita delle due logiche: elementare per la prima, polinomiale per la seconda.

Grazie all'uso di tali logiche come sistemi di tipo per il λ calcolo, la presenza dell'oracolo, che effettua le operazioni di manutenzione degli indici, risulta inutile, quindi il lavoro di Baillot, Coppola e Dal Lago si è potuto concentrare sull'analisi di complessità dell'algoritmo di riduzione astratto.

Anche questo risultato del loro lavoro, viene ottenuto per mezzo della semantica dei contesti, usata in maniera "quantititiva". Attraverso la tecnica recentemente introdotta dallo stesso Dal Lago (2009), gli autori definiscono una nozione di *peso* W dei grafi di condivisione, che viene inizialmente limitato grazie ai risultati classici della complessità delle logiche. È noto, dal teorema 5.1, infatti, che per ogni riduzione di EAL (LAL), la lunghezza della riduzione e la taglia della rete di prova ridotta siano limitate da una funzione elementare (polinomiale). Gli autori possono limitare la complessità delle riduzioni condivise in funzione dello stesso dei grafi, ottenendo il seguente risultato, nel quale $\mathcal{T}$ è la relazione che indica ogni traduzione ammissibile da λ termini tipati a grafi di condivisione.

Teorema 4.4 (Complessità dell'implementazione condivisa di $\Lambda_{EAL/LAL}$)**.** *Per ogni naturale d, esiste una funzione elementare (rispettivamente polinomiale) f_d tale che, per ogni rete rete di prova N di EAL (LAL), con $\partial(N) = d$, e per la quale $N \xrightarrow{r}_{EAL} N'$; esistono un grafo di condivisione G tale che $\mathcal{T}(N) = G$ e una riduzione condivisa $G \xrightarrow{s}_{SG} G'$ per la quale $|s| \leq f_{\partial(N)}(|r|)$.*

La dimostrazione, attraverso l'approccio derivante dalla geometria d'interazione, consiste in una minuziosa analisi dei percorsi presenti nei grafi di condivisione e di come questi evolvano lungo la loro riduzione. Nonostante la tecnica sia uno strumento interessante per l'analisi della complessità computazionale, appare eccessivamente complesso per il caso trattato.

La motivazione principale che ha indirizzato gli sforzi di questa tesi di Laurea Magistrale è stata proprio quella di fornire una diversa dimostrazione di questo risultato per diretta via sintattica, situandosi in un'impostazione di lavoro precedentemente sviluppata da Guerrini *ed altri* (2000, 2003). Lavorando sul ristretto caso privo di oracolo, infatti, è stato possibile, sfruttare l'efficace tecnica di simulazione anche per dimostrare il limite di complessità alla riduzione condivisa, che è presentata nei successivi capitoli 6 e 7.

Che la riduzione ottimale abbia complessità davvero ottimale resta un problema, ad oggi, ancora aperto.

5. Logiche affini

> Nessuno saggia la profondità di un fiume con entrambi i piedi.
>
> Proverbio africano

In questo capitolo presenteremo le logiche affini elementare e leggera a partire da una breve panoramica del contesto che ne ha visto la nascita e la crescita negli ultimi due decenni. Sarà presentata una definizione formale di questi due sistemi logici per mezzo di un sistema deduttivo a sequenti e di reti di prova, e per mezzo di quest'ultima formulazione, sarà brevemente illustrata una relazione di normalizzazione. Le sezioni 5.3 e 5.4 enunceranno la proprietà di stratificazione e i risultati di complessità per EAL e LAL, mentre quella conclusiva discuterà della ignorata presenza di spazzatura nelle reti di prova.

5.1. Introduzione

L'intento di riconciliare la simmetrica eleganza della logica classica con la costruttiva utilità di quella intuizionista portò Girard alla formulazione della logica lineare Girard (1987). Il sistema permise un nuovo punto di vista sul contenuto computazionale delle logiche, per mezzo del quale le formule possono essere trattate come risorse, consumabili e non necessariamente infinite, ed ha pertanto offerto, e continua ad offrire, numerosi spunti al progresso della scienza computazionale.

Oltre a consentire la pulita formulazione dell'implementazione ottimale presentata nel capitolo 3, ha fornito nuovi strumenti alla ricerca nella complessità implicita, che mira alla caratterizzazione intrinseca delle classi di complessità computazionale. Tale indirizzo di studi [1] si era inizialmente focalizzato sulla differenziazione degli oggetti computazionali, per riuscire a limitare la complessità della ricorsione che opera su di essi: la ricorsione sicura e quella stratificata Bellantoni e Cook (1992); Leivant (1994). Lo stesso Girard Girard (1995) propose poi un sistema logico intrinsecamente polinomiale, cioè corretto e completo rispetto alla classe di algoritmi relativa: la logica lineare leggera (LLL). Il sistema deduttivo possiede una nuova modalità, il

[1] Per una rassegna approfondita, seppure non troppo recente, sui progressi della complessità computazionale implicita, si rimanda a Hofmann (2000).

§, che permette il controllo della regola di contrazione, responsabile della potenziale esplosione di complessità.

La LLL venne successivamente semplificata da Asperti nella logica affine leggera (LAL) Asperti (1998), permettendo l'uso completo della regola di indebolimento, senza intaccarne le interessanti proprietà computazionali. Più recentemente gli studi sono continuati sulla variante elementare della logica affine (EAL) e soprattutto sulle relazioni con la programmazione funzionale, offerta dai relativi frammenti intuizionistici, in particolare ILAL.

5.2. Definizione

5.2.1. Sistema deduttivo

Le logiche affini che sono state prese in considerazione sono quella elementare (EAL) e quella leggera (LAL), nella versione al second'ordine e con operatore di punto fisso. L'operatore $\forall$ d'ordine superiore sostituisce il frammento additivo della LL classica, mentre il punto fisso risulta conveniente per l'uso della logica come sistema di tipi, dato che esso ne aumenta l'espressività premettendo l'uso di tipi ricorsivi.

Definizione 5.1 (Formule)**.** L'insieme $\mathcal{F}_{EAL}$ di formule di EAL sono generate dalla grammatica con le seguenti regole di produzione, dove α indica una variabile fra un insieme enumerabile di variabili.

$$A ::= \alpha \mid A \multimap A \mid !A \mid \forall \alpha.A \mid \mu\alpha.A$$

L'insieme $\mathcal{F}_{LAL}$ di formule di LAL sono generate dalla regole di produzione di EAL unite a:

$$A ::= \S A$$

Il λ calcolo puro offre, da un punto di vista statico, la sintassi delle prove di una formula delle logiche affini, ovvero i termini da tipare in tali logiche.

Definizione 5.2 (Termini)**.** I termini Λ sono i termini del λ calcolo puro, definito dalle seguenti produzioni, dove x indica una variabile fra un insieme enumerabile di variabili.

$$t, u ::= x \mid \lambda x.t \mid tu$$

Definizione 5.3 (Giudizi e sequenti)**.** Un *giudizio* è una coppia $t : A$, dove t è un termine in Λ e A è una formula di $\mathcal{F}_{EAL}$ o $\mathcal{F}_{LAL}$. Un *sequente* è formato da un insieme di giudizi negativi ed un giudizio positivo, espresso nella forma $\Gamma \vdash t : A$.

Le regole per stabilire giudizi di tipo costituiscono il calcolo a sequenti per le logiche affini elementare e leggera. Useremo le lettere minuscole della seconda parte dell'alfabeto per i termini di Λ e quelle finali per le relative variabili, le maiuscole

$$\frac{}{x : A \vdash x : A}(A) \qquad \frac{\Gamma \vdash t : A \quad \Delta, x : A \vdash u : B}{\Gamma, \Delta \vdash u[{}^{t}/_{x}] : B}(U)$$

$$\frac{\Gamma \vdash t : B}{\Gamma, x : A \vdash t : B}(W) \qquad \frac{\Gamma, x : !A, y : !A \vdash t : B}{\Gamma, z : !A \vdash t[{}^{z}/_{x}, {}^{z}/_{y}] : B}(C)$$

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x.t : A \multimap B}(R_{\multimap}) \qquad \frac{\Gamma, t : \ A \quad \Delta, x : B \vdash u : C}{\Gamma, \Delta, y : A \multimap B \vdash u[{}^{yt}/_{x}] : C}(L_{\multimap})$$

$$\frac{\Gamma \vdash t : A \quad \alpha \notin FV(\Gamma)}{\Gamma \vdash t : \forall \alpha.A}(R_{\forall}) \qquad \frac{\Gamma, x : A[{}^{B}/_{\alpha}] \vdash t : C}{\Gamma, x : \forall \alpha.A \vdash t : C}(L_{\forall})$$

$$\frac{\Gamma \vdash t : A[{}^{\mu\alpha.A}/_{\alpha}]}{\Gamma \vdash t : \mu\alpha.A}(R_{\mu}) \qquad \frac{\Gamma, x : A[{}^{\mu\alpha.A}/_{\alpha}] \vdash t : B}{\Gamma, x : \mu\alpha.A \vdash t : B}(L_{\mu})$$

Figura 5.1.: Calcolo a sequenti per logiche affini: frammento comune

$$\frac{\Gamma \vdash t : A}{!\Gamma \vdash t : !A}(!)$$

Figura 5.2.: Calcolo a sequenti per logiche affini: esponenziale per EAL

della prima parte dell'alfabeto per indicare formule in $\mathcal{F}_{EAL}$ o $\mathcal{F}_{LAL}$, minuscole greche per le relative variabili, e maiuscole greche per insiemi di giudizi, detti anche contesti. Inoltre, l'applicazione di esponenziali a contesti ne indica la distribuzione alle formule in essi contenuti.

Definizione 5.4 (Enunciati)**.** Gli enunciati delle logiche affini sono definiti dal calcolo a sequenti della figura 5.1, cui aggiungere le regole relative agli esponenziali: figura 5.2 per EAL e figura 5.3 per LAL.

La parte interessante del sistema risiede nei connettivi di modalità e le relative regole, attraverso i quali viene gestito il comportamento non lineare della logica e, quindi, la complessità di riduzione. Oltre alla modalità ! (*of course*) della logica lineare, LAL introduce una stretta disciplina per la sua introduzione ed una seconda

$$\frac{\vdash t : A}{\vdash t : !A}(!_{l}) \qquad \frac{x : A \vdash t : A}{x : !A \vdash t : !A}(!_{b}) \qquad \frac{\Gamma, \Delta \vdash t : A}{\S\Gamma, !\Delta \vdash t : \S A}(\S)$$

Figura 5.3.: Calcolo a sequenti per logiche affini: esponenziali per LAL

modalità § (*paragraph*) sulla quale non è possibile contrarre. In tal modo, la contrazione non è più applicabile in LAL, dato che essa opera su formule ! negative, che in LAL possono essere introdotte solo una per volta. Nelle reti di prova, le scatole di tipo ! hanno, insomma, esattamente una porta secondaria (cfr. fig. 5.6).

5.2.2. Inclusione di LAL in EAL

È facile notare come il sistema LAL sia a tutti gli effetti un sottoinsieme di EAL. Ci basta considerare una funzione $\mathcal{E} :: \mathcal{F}_{LAL} \mapsto \mathcal{F}_{EAL}$, costruita affinché sostituisca ogni occorrenza di § nelle formule LAL con !.

Proposizione 5.1 (Inclusione di LAL in EAL)**.** *Sia $t \in \Lambda_{LAL}$ un termine, per cui cioè esiste un contesto Γ e una formula A tale che $\Gamma \vdash t : A$. Allora $t \in \Lambda_{EAL}$, ovvero esiste Δ per cui $\Delta \vdash t : \mathcal{E}(A)$.*

Tale rapporto si estende anche alle reti di prova e alle regole di normalizzazione, che saranno quindi presentate per la sola EAL, il sistema più espressivo dei due. In generale, nel seguito della trattazione faremo, per semplicità, riferimento a LAL solo quando si rende necessaria una differenziazione.

5.2.3. Reti di prova

Le reti di prova sono un formalismo che esprime geometricamente e sinteticamente le deduzioni di un certo sistema e le relazioni di riscrittura per l'eliminazione del taglio. In esse tutta la profondità e la bellezza della corrispondenza di Curry–Howard si manifesta, riuscendo ad esplicitare la relazione fra λ termini e formule nella sua completezza.

Da un punto di vista strutturale, le rete di prova sono ipergrafi, orientati rispetto alla direzione premesse - conclusioni, i cui ipernodi sono formule e i cui iperarchi sono collegamenti fra le formule. I collegamenti possibili fra formule corrispondono, uno ad uno, alle regole del sistema deduttivo; saranno rappresentati per mezzo di vertici cerchiati, eccetto per le regole dell'assioma e del taglio.

La definizione classica delle reti di prova di EAL e LAL, consiste nello stabilire quali siano le *strutture di prova*, induttivamente componibili a partire dalle formule di EAL e dai vertici. Graficamente le formule saranno indicate come etichette sui rami dei collegamenti. La differenziazione dei rami fra principali e secondari differenzia la direzione nella quale il collegamento opera, mentre la positività differenzia premesse e conclusioni della prova.

Definizione 5.5 (Vertici)**.** I vertici delle strutture di prova sono i seguenti.

- unari, con un ramo negativo: $\{(W)\}$
- binari, con un ramo negativo ed uno positivo principale: $\{(R_!), (R_\forall), (R_\mu)\}$

- binari, con un ramo negativo ed uno positivo principale: $\{(L_!), (L_\vee), (L_\mu),\}$
- ternari, due rami negativi ed uno positivo principale: $\{(R_{\multimap})\}$
- ternari, un ramo positivo principale e due negativi: $\{(L_{\multimap}), (C)\}$

Per poter isolare le sole strutture di prova che siano aderenti al calcolo a sequenti di EAL è, a questo punto, necessario imporre ulteriori condizioni alla formazione delle strutture, un criterio di correttezza. Una formulazione più semplice ed elegante rispetto all'originale di Girard consiste in una condizione di aciclicità su percorsi diretti.

Definizione 5.6 (Percorso diretto)**.** Un percorso è una sequenza di collegamenti $c_1, \ldots, c_n$ tali che per ogni $1 \leq i \leq n$, i collegamenti c_i ed c_{i+1} sono diversi fra loro, ed hanno un nodo comune n_i per il quale uno dei due è collegamento principale.

Definizione 5.7 (Criterio di correttezza Danos e Regnier (1989))**.** Una struttura di prova è una rete di prova se essa non contiene percorsi diretti ciclici.

Il criterio di correttezza è la condizione sufficiente affinché una struttura di prova sia effettivamente una rete di prova.

Definizione 5.8 (Reti di prova)**.** Le *reti di prova* EAL o LAL sono tutte quelle strutture di prova che soddisfano il criterio di correttezza.

Le reti di prova delle logiche affini possono, però, essere definite in maniera visiva e sperabilmente più comprensibile, massimizzando, peraltro, la corrispondenza rispetto al calcolo a sequenti 5.4). Le figure 5.4, 5.5, 5.6, 5.7 presentano le regole di formazione di reti di prova, da leggersi nello stile premessa - conclusione: la prima è tracciata in linee solide, la seconda è invece compresa dalle linee tratteggiate dagli eventuali vertici ad esse connessi.

Si tenga presente, inoltre, che, secondo la notazione utilizzata, i rettangoli arrotondati N, M indicano reti di prova, che le porte principali dei vertici sono indicati col simbolo $\bullet$, e che le sottoreti circondate da esponenziali sono delimitate da una decorazione chiamata scatola, rappresentata dal rettangolo a linea spessa.

5.2.4. Eliminazione del taglio

L'eliminazione del taglio da una prova, che da un punto di vista logico ne costituisce il processo di normalizzazione, corrisponde, tramite la corrispondenza di Curry-Howard, alla computazione del λ termine. Per la logica EAL (e quindi LAL) essa è formalizzabile attraverso un sistema di riscrittura su grafi, cioè sulle reti di prova.

Definizione 5.9 (Eliminazione del taglio)**.** La relazione $\rightarrow_{EAL}$, è la più piccola relazione che contiene le coppie indicate nelle figure 5.8, 5.9, 5.10, 5.11.

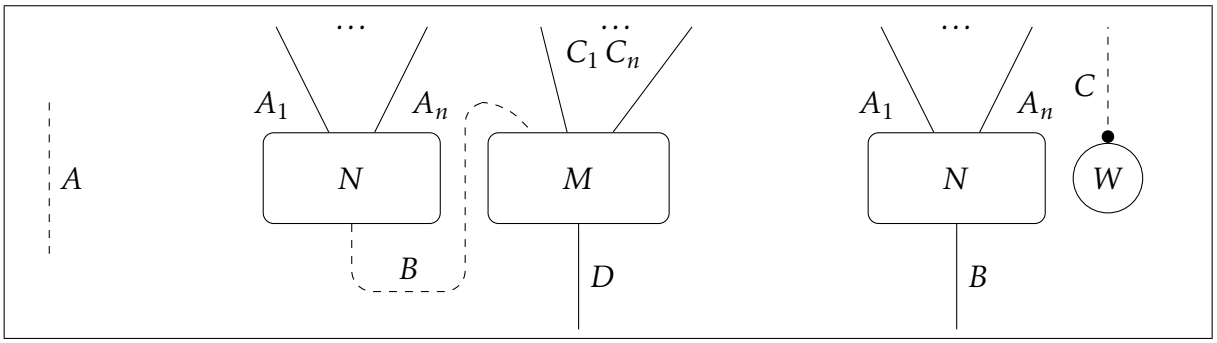


Figura 5.4.: Reti di prova per logiche affini: assioma, taglio e indebolimento

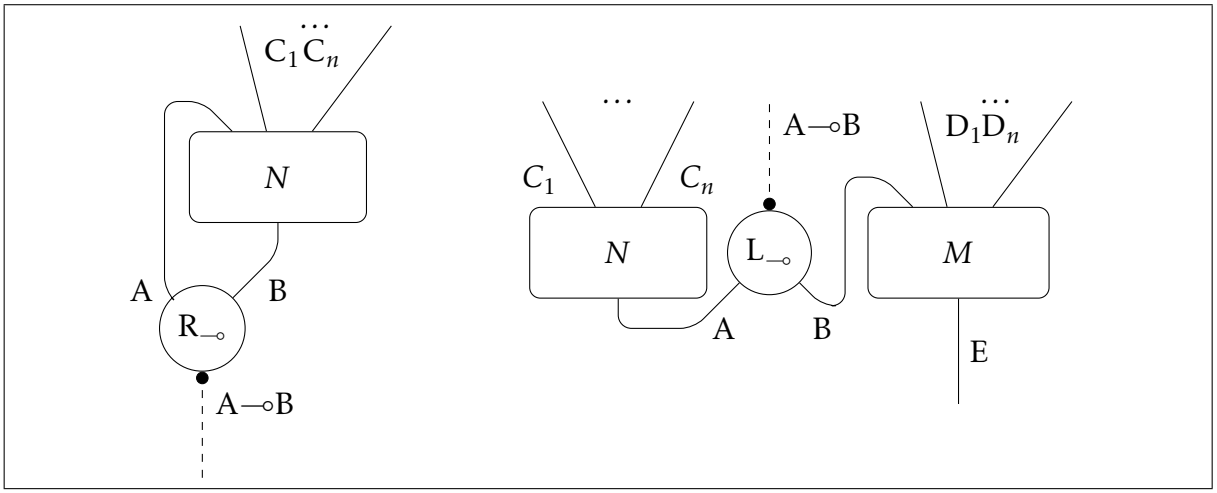


Figura 5.5.: Reti di prova per logiche affini: implicazione

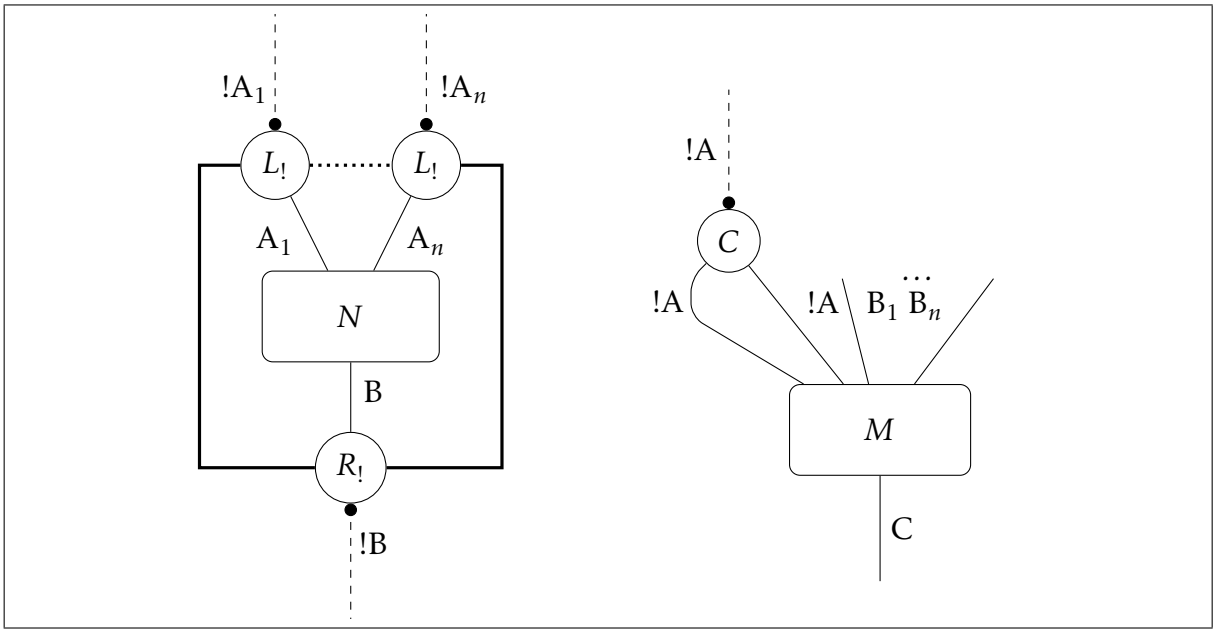


Figura 5.6.: Reti di prova per logiche affini: esponenziale e contrazione

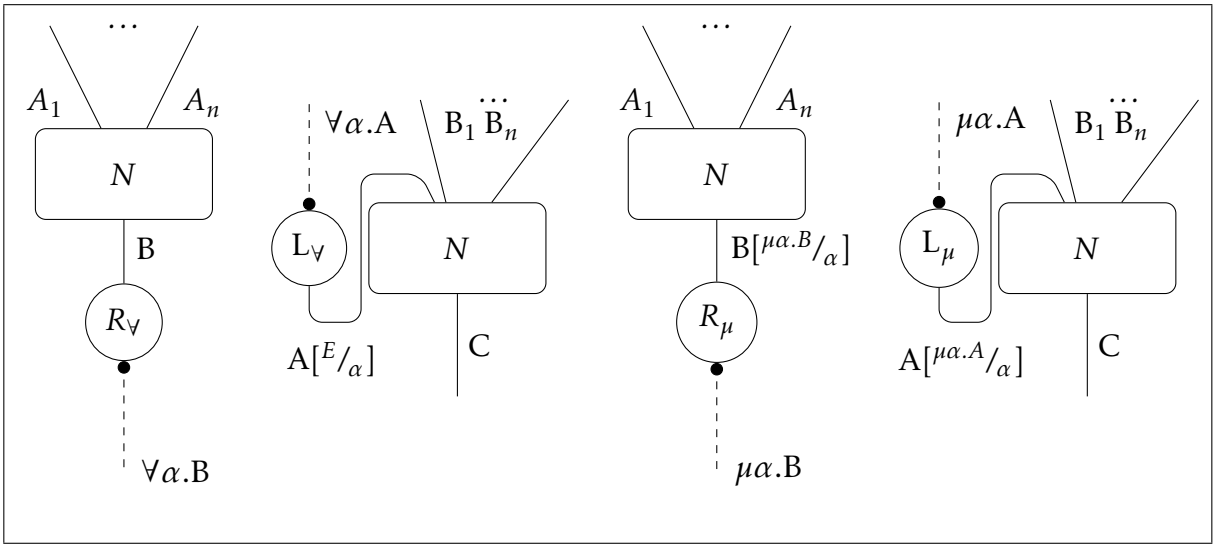


Figura 5.7.: Reti di prova per logiche affini: second'ordine e punto fisso

Le regole non includono, per semplicità la riduzione di collegamenti di indebolimento (W), come spiegato dalla successiva sezione 5.5.

Come consueto, $\rightarrow^+_{EAL}$ indicherà la relazione che è chiusura transitiva di $\rightarrow_{EAL}$, mentre $\rightarrow^*_{EAL}$ quella transitiva e riflessiva. Inoltre, la notazione $\xrightarrow{r}_{EAL}$, fa riferimento ad r come una specifica istanza di $\rightarrow^*_{EAL}$.

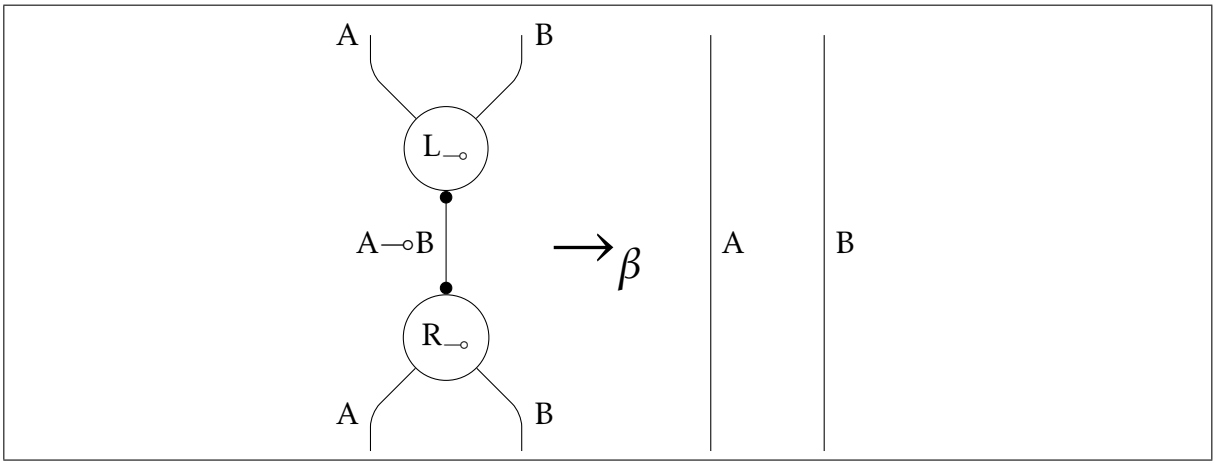


Figura 5.8.: Eliminazione del taglio per logiche affini: β riduzione

5.3. Stratificazione

Dalle regole di formazione delle reti di prova, risulta chiaro che le scatole sono sovrapponibili, ma solo creando annidamenti completamente ordinati, non è possibile

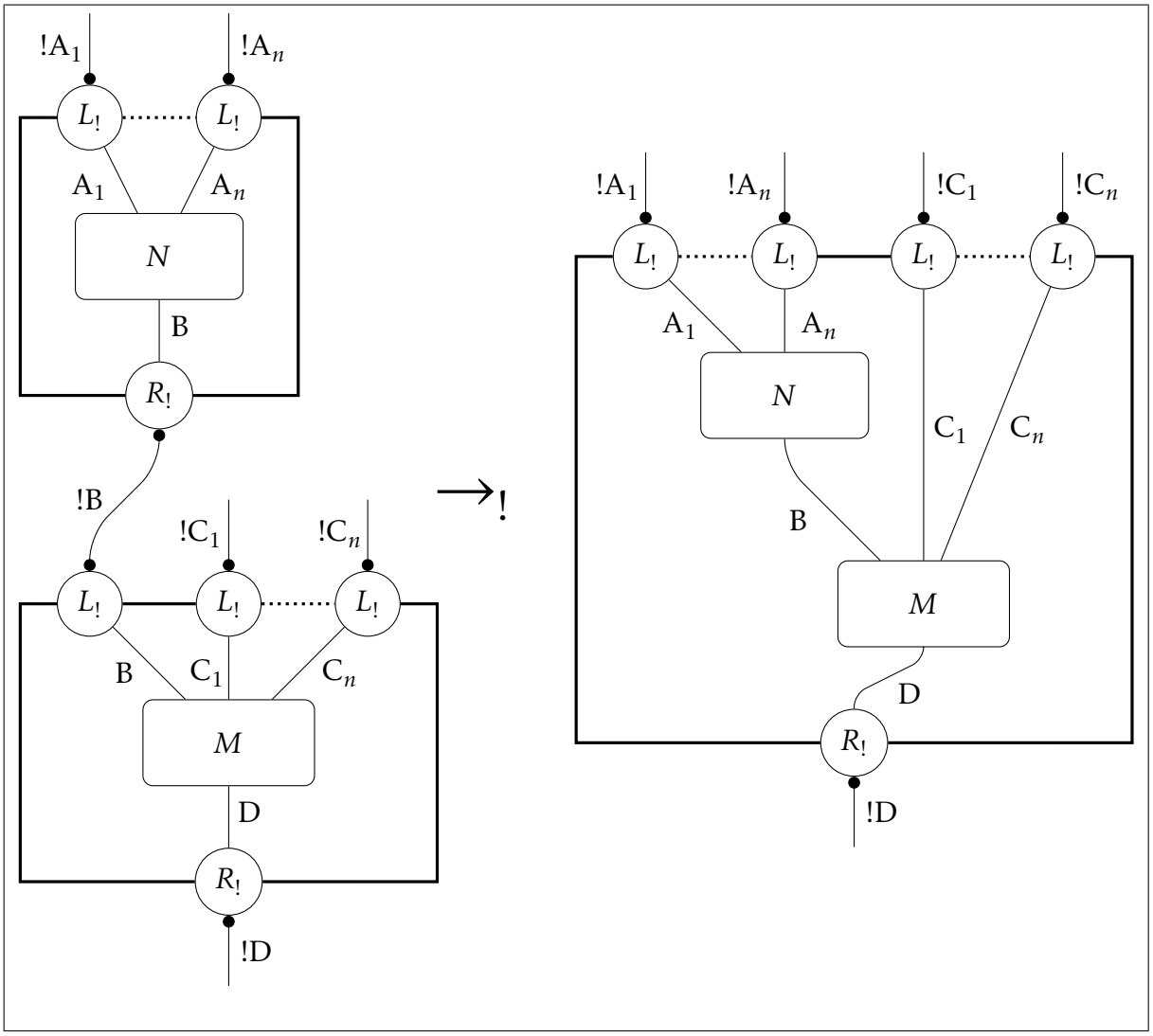


Figura 5.9.: Eliminazione del taglio per logiche affini: fusione di scatole

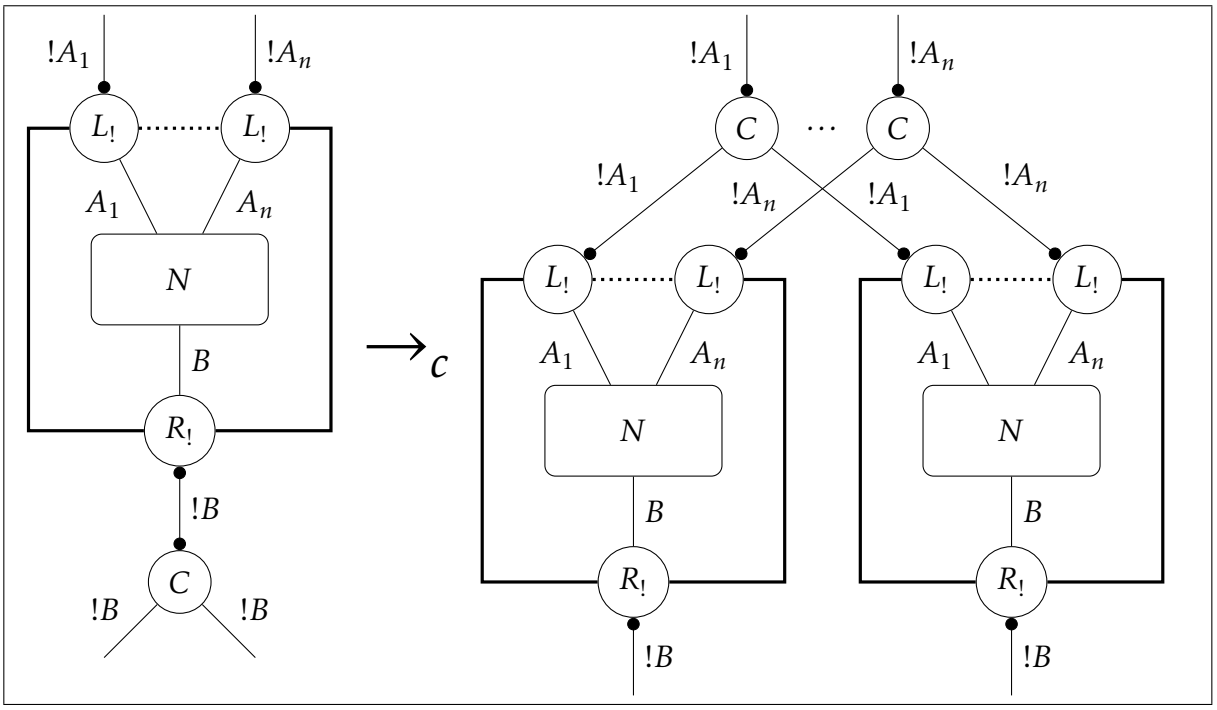


Figura 5.10.: Eliminazione del taglio per logiche affini: duplicazione

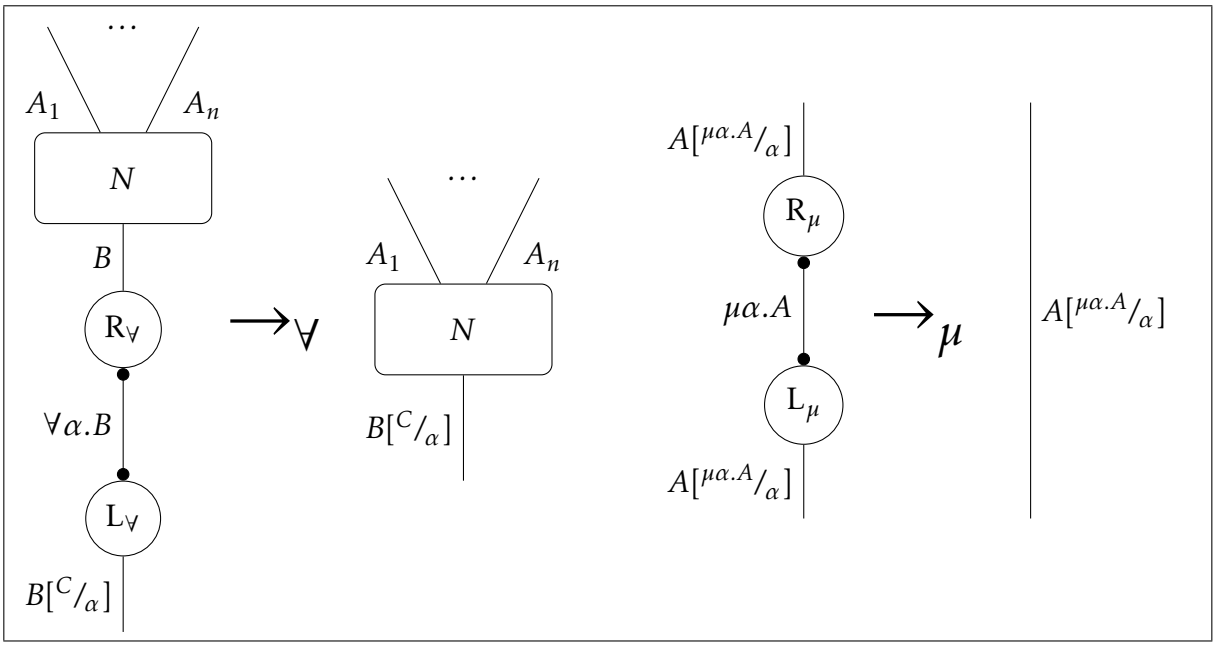


Figura 5.11.: Eliminazione del taglio per logiche affini: second'ordine e punto fisso

cioè che che se un vertice appartiene a due scatole, la prima non sia strettamente inclusa nella seconda o viceversa.

Le logiche non limitate, come MELL, che è la logica lineare ristretta ai frammenti moltiplicativo, dei connettivi $\otimes$ e $\parr$, ed esponenziale, hanno una gestione più lasca della relativa modalità. In particolare, i collegamenti esponenziali possono attraversare più scatole e quindi i vertici ($L_!$) non sono strettamente legati ad esse come nelle reti di prova di EAL (cfr. fig. 5.6). In questi casi, pertanto, la regola riscrittura di fusione di scatole, che elimina un taglio esponenziale (fig. 5.9), è in grado di spostare scatole a livelli di annidamento differenti.

Definiamo dunque in maniera formale il concetto di livello di annidamento per nodi e reti di prova.

Definizione 5.10 (Profondità)**.** Un vertice n appartenente ad una rete di prova N è a profondità l, detta anche livello e scritta $\partial(n)$, se è contenuto in l scatole. La massima profondità di N, denotata con $\partial(N)$ è il livello massimo dei suoi nodi.

Per le logiche limitate qui considerate vale un'importante proprietà: l'invarianza della profondità di un vertice è rispetto all'eliminazione del taglio.

Proposizione 5.2 (Stratificazione)**.** *Presa una rete di prova N di EAL, sia $n \in N$ un vertice, sia $N \longrightarrow_{EAL} N'$ una riduzione e sia $\{n_1, \dots n_k \in N'\}$ l'insieme (eventualmente vuoto) delle immagini della riduzione. Allora per ogni n_i, con $i \leq k$ vale che $\partial(n_i) = \partial(n)$.*

Questa proprietà semplificherà notevolmente il sistema di riduzione condivisa su reti EAL che sarà presentato nel seguente capitolo 7.

5.4. Complessità

Nella precedente sezione 5.1 introduttiva al presente capitolo si è accennato al fatto che le logiche affini qui presentate esprimono intrinsecamente classi di complessità interessante e a qualcuno dei relativi contributori.

La logica lineare leggera venne presentata da Girard in Girard (1995), dimostrandone la correttezza rispetto alla classe polinomiale. Considerata una strategia di riduzione delle reti intuitivamente pessima, che quindi riduce a livello di profondità via via crescente, e considerando come costi della normalizzazione sia la sua lunghezza che la variazione di dimensione della rete, questi sono limitati in funzione della dimensione originale della rete. In Asperti e Roversi (2002) Asperti e Roversi riassumono il loro precedente lavoro di semplificazione del sistema logico nella variante affine LAL, di studio del relativo frammento intuizionista ILAL e della dimostrazione di completezza rispetto alla classe Ptime: ogni computazione di una macchina di Turing polinomiale è simulabile per mezzo di una riduzione di tale sistema. Più recentemente Terui ha stabilito in Terui (2007) l'atteso risultato di correttezza forte: la complessità è ben limitata per ogni strategia di riduzione.

Danos e Joinet si occupano, in Danos e Joinet (2003), della caratterizzazione della classe elementare, che è unione della gerarchia esponenziale, per mezzo di ELL, un'ulteriore variazione della ILL, frammento intuizionistico della logica lineare.

Per riassumere formalmente il risultato di complessità per la riduzione delle logiche EAL e LAL, introduciamo una necessaria definizione.

Definizione 5.11 (Dimensione)**.** La dimensione $|N|$ di una rete di prova N è il numero dei suoi collegamenti.

Definizione 5.12 (Funzione elementare)**.** Una funzione è *elementare* se esiste un naturale i per cui appartiene a $\mathcal{O}(K(i,n))$, dove K è la funzione di Kalmar così definita.

$$K(i,n) = \begin{cases} n & se\ i = 0 \\ K(i-1,n) & se\ i > 0 \end{cases}$$

Teorema 5.1 (Correttezza forte della complessità)**.** *Per ogni naturale d esiste una funzione elementare (rispettivamente polinomiale) f_d tale che per ogni rete di prova N in EAL (LAL) per la quale $\partial(N) = d$, se $N \xrightarrow{r}_{ED} N'$, allora*

- $|r| \leq f_d(|N|)$,
- $|N'| \leq g_d(|N|)$.

Questo risultato costituisce una pietra angolare della trattazione successiva, visto che la limitazione della complessità dell'implementazione condivisa sarà dimostrata proprio grazie alla limitazione nella lunghezza delle riduzioni EAL e LAL.

5.5. Spazzatura

La regola di indebolimento è stata presentata nella versione lasca delle logiche affini con la regola (W) nel calcolo in figura 5.1 e nelle reti di prova in figura 5.4). Tale regola, anche nel caso della versione rigida delle logiche lineari, che impone la modalità non lineare alle formule introdotte, consente l'indesiderata presenza di sottoreti spazzatura. A partire dalla formula che essa introdotta è possibile infatti accrescere la rete di prova verso l'alto dando origine a porzioni di prova superflue.

Il processo di normalizzazione della prova può pertanto includere delle regole di raccoglimento della spazzatura, che eliminino queste sottoreti. Il costo dell'operazione è chiaramente lineare nella dimensione della spazzatura e al peggio lineare nella dimensione dell'intera rete. Dato che una rete ridotta EAL (rispettivamente LAL) ha dimensione al peggio elementare (polinomiale) nella dimensione della rete originale, anche il costo di raccoglimento della spazzatura è ben limitato e, quindi, sicuramente ignorabile.

Le regole di normalizzazione introdotte, pertanto, trascurano le regole di raccoglimento della spazzatura.

6. Reti a duplicazione esplicita

> We think in generalities,
> but we live in detail.
>
> Alfred North Whitehead
> (1861-1947)

In questo capitolo sarà introdotto un sistema di riscrittura intermedio fra le reti di prova delle logiche affini e i grafi di condivisione per tali reti, che getta luce formale sulle principali differenze e relazioni fra i due. In particolare, nella riduzione su grafi di condivisione la duplicazione avviene in maniera atomica, ovvero un nodo alla volta, mentre la riduzione su reti di prova opera per grandi passi, cioè una scatola alla volta. Il nuovo sistema di riscrittura, definito come estensione delle regole d'eliminazione del taglio su EAL o LAL, introduce quindi dei marcatori che hanno il compito di fluire lungo le sottoreti duplicate, in modo da tradurre il costo di duplicazione in passi di riscrittura espliciti.

Come sarà chiaro nel presente e nel successivo capitolo, il sistema ED è il risultato di un processo di definizione delicato, in equilibrio fra i due sistemi di riscrittura fra i quali si pone. Procederemo dapprima individuando alcuni degli aspetti che ne hanno influenzato la forma qui presentata, e solo poi affrontando completamente e formalmente la sua definizione e le sue proprietà. Infine, nella sezione finale, sarà dimostrato il limite di complessità nella lunghezza della riduzione delle reti a duplicazione esplicita, riducendolo alla complessità delle corrispondenti reti di prova, che costituirà un primo passo per la dimostrazione del risultato principale sulla complessità dei grafi di condivisione che ci attende, invece, nel successivo capitolo 7.

6.1. Motivazioni

La regola di eliminazione del taglio per nodi di contrazione dei sistemi EAL e LAL permette di circoscrivere puntualmente l'origine della complessità di un sistema computazionale, come abbiamo visto nella precedente sezione 5.4, ma nasconde in un passo un'operazione affatto unitaria. Prendiamo ad esempio la figura 6.1: un nodo (C), supponiamo n-ario, è collegato alla porta principale di una scatola S con m altre porte secondarie. La riscrittura porta il contrattore, in un solo passo, oltre la

scatola, sostituita da n occorrenze, ed esso stesso è moltiplicato in m istanze, ognuna collegata ad ognuna delle scatole. Il costo del passo di riscrittura potrebbe essere insomma stimato in $n(|S| + m)$.

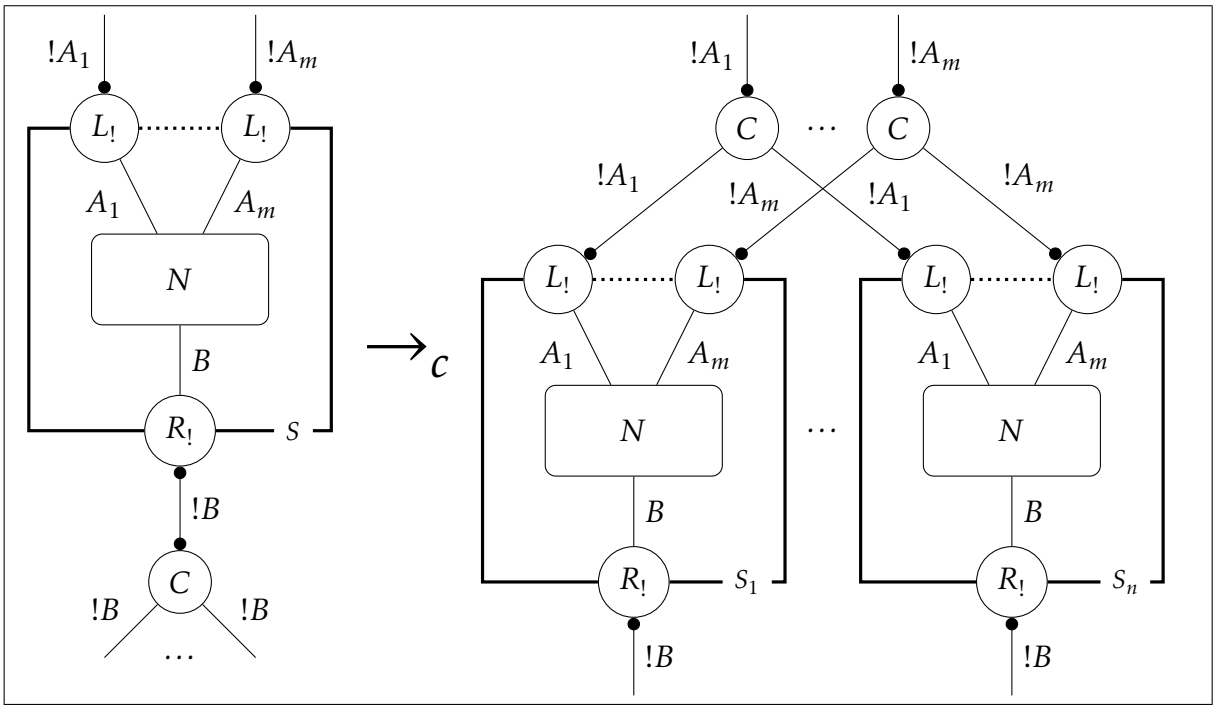


Figura 6.1.: Esempio di duplicazione nelle logiche affini

La naturale corrispondenza fra la contrazione delle reti di prova e la condivisione dell'implementazione ottimale, fra i nodi (C) della prima e i fan della seconda, risulta pertanto debolmente espressa, visto che quest'ultima si esprime in stile assolutamente atomico. Volendo esplicitare il celato lavoro di duplicazione, l'idea da cui si è partiti è quella di introdurre un nodo non logico nel sistema di riscrittura, che trasformi tale costo in passi di riduzione facendolo fluire attraverso le copie generate in tale operazione.

6.1.1. Marcatori di duplicazione

Il punto di partenza è quindi una modifica alla regola di duplicazione, affinché essa aggiunga un nodo marcatore, chiamato lift e dall'aspetto che richiama un nodo mux, all'entrata di ognuna delle porte principali delle copie della scatola.

Il lift hanno il compito di fluire nelle sottoreti interne alle copie, propagandosi attraverso ognuno dei nodi, e dal punto di vista strettamente logico sono del tutto trasparenti, dato che i suoi collegamenti hanno tipi identici, che cambiano a seconda del nodo che attraversano. La figura 6.2 mostra la regola di duplicazione così modificata, della quale si può ignorare, per ora, la decorazione al nodo C.

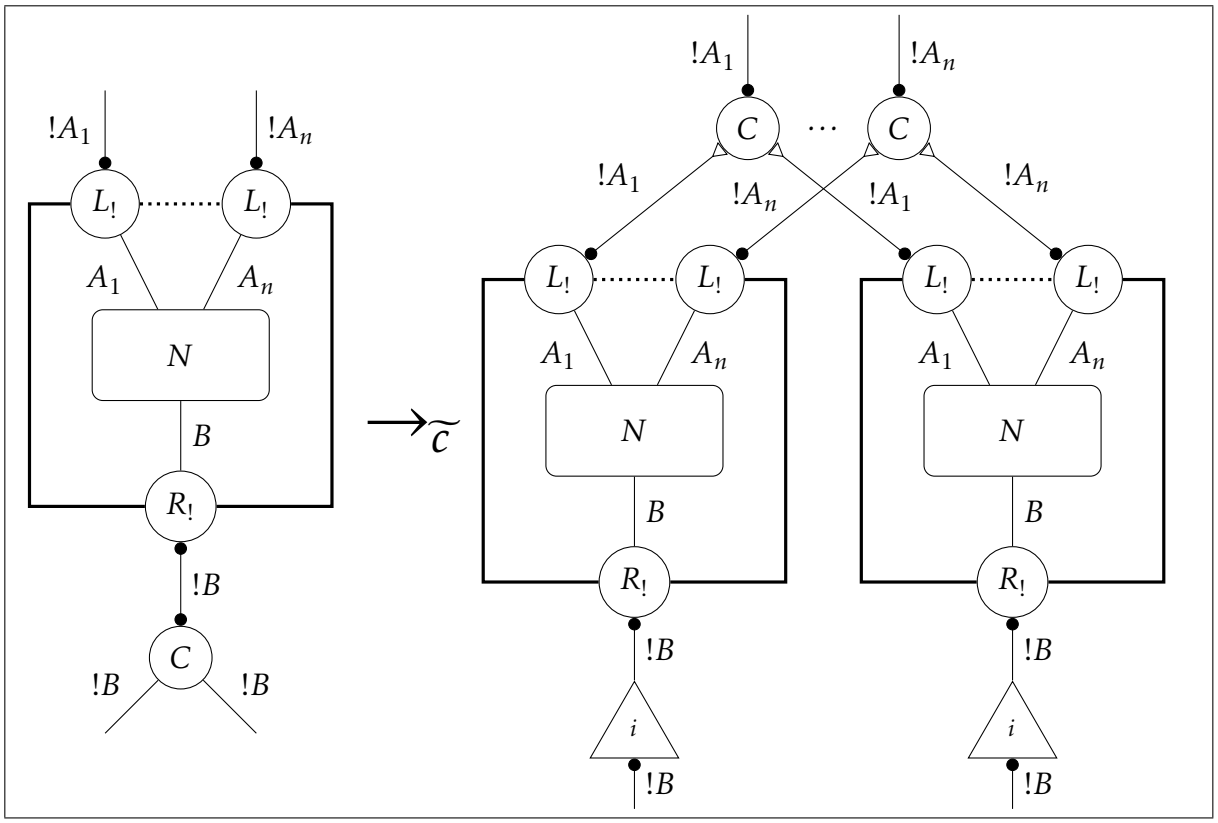


Figura 6.2.: Riscrittura a duplicazione esplicita: duplicazione

6.1.2. Propagazione dei lift

Altrettanto intuitiva è la descrizione formale della regola di avanzamento di un lift, ovvero di come esso scorra i nodi logici. Nella figura 6.3 sono mostrate le due regole di avanzamento del contrattore, una attraverso un nodo binario, l'altra uno ternario. Si ignori, per il momento, gli indici i, j e la condizione su di essi.

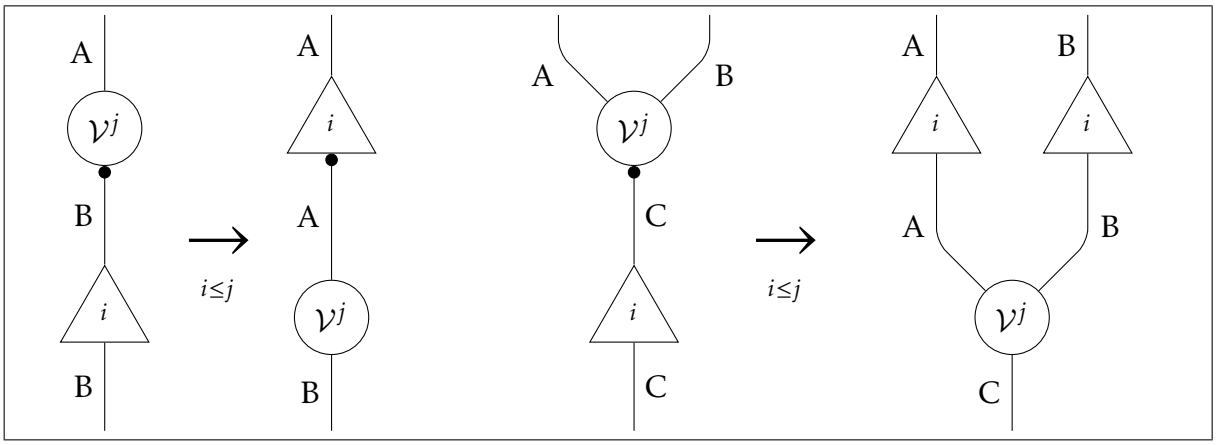


Figura 6.3.: Riscrittura a duplicazione esplicita: avanzamento dei lift su nodi binari e ternari

La propagazione di un lift immaginiamo abbia termine quando esso raggiunga, oltre la porta secondaria della scatola che sta marcando, il contrattore che lo aveva generato, o quando esso incontri un altro lift, col quale abbia cooperato nel portare a termine la propagazione attraverso una sottorete. Ma, in generale, nella rete potrebbero essere attivi numerosi lift, anche relativi a diverse duplicazioni di scatole annidate, quindi ci aspettiamo che qualora due lift si incontrino i comportamenti possibili siano due: l'annichilamento e lo scambio.

Per poter distinguere i due casi, la soluzione migliore, di nuovo ispirata ai grafo di condivisione è l'indicizzazione di ogni lift a seconda del livello di profondità della scatola su cui sta lavorando: se corrisponde, i due lift si annullano l'un l'altro, altrimenti si scambiano. La figura 6.4 mostra i due comportamenti e le relative regole a margine sugli indici dei lift. Si noti che tale definizione risulta ben posta grazie alla stratificazione (proposizione 5.2) che è garantita dalle reti delle logiche leggere EAL e LAL: il livello di profondità è invariante durante la riduzione, quindi lo sono anche gli indici assegnati ai lift.

Confrontando il comportamento dei lift con quello dei fan dei grafi di condivisione, di cui si è trattato nella sezione 3 si può notare come il comportamento dei primi sia stato reso in maniera molto simile a quello dei secondi. Il lettore esperto di implementazione ottimale avrà già capito che il nome dei lift è proprio un prestito dai condivisori unari, usati per la correlazione con λ-alberi Asperti e Guerrini (1998); Guerrini (1999).

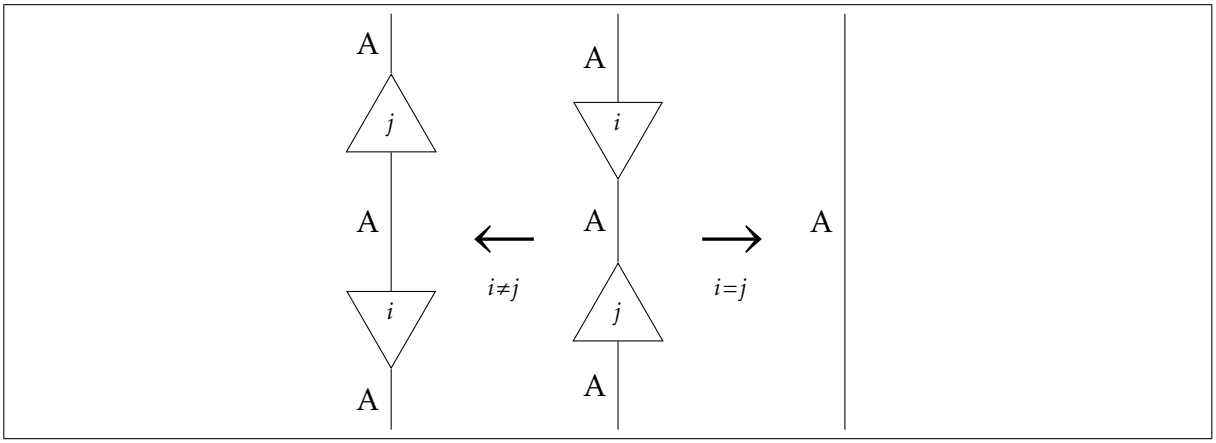


Figura 6.4.: Riscrittura a duplicazione esplicita: incontro fra lift, scambio o annichilamento

6.1.3. Arresto dei lift

Per gestire la terminazione del lavoro di marcatura quando un lift esce dalla porta secondaria della scatola duplicata, vorremmo che fosse semplicemente e naturalmente assorbito dal nodo contrattore che lo ha generato. Ma tale contrattore potrebbe voler essere ridotto, eseguendo ulteriori duplicazioni di diverse scatole.

Possiamo inoltre notare che, qualora una scatola con lift in azione si trovi ad essere direttamente collegata ad un'altra per mezzo di un taglio esponenziale, ovvero fra due nodi di tipo $L_!$ e $R_!$, questo taglio potrebbe essere eliminato. Il che porterebbe alla fusione delle due scatole, ed alla indebita marcatura, da parte dei lift attivi nella prima, di nodi non afferenti alla loro propria duplicazione.

La soluzione più ovvia per risolvere in un sol colpo entrambi i problemi sarebbe quella di imporre una strategia di riduzione, che dia priorità assoluta all'insieme di azioni π, che realizzano la propagazione dei lift. Se la marcatura costituisse la prima necessaria parte di ogni riduzione che segue una duplicazione, potremmo infatti essere certi che il contrattore resti fermo ad attendere i lift in uscita e che nessuna fusione abbia luogo. Ma tale modifica avrebbe però ripercussioni sulla corrispondenza con i grafi di condivisione, giacché forzerebbe una specifica strategia anche sulla riduzione condivisa, eliminando proprio quella ottimale. E nemmeno possiamo introdurre un ulteriore nodo ausiliario all'uscita delle scatole, in ognuna delle porte secondarie, che resti in attesa dell'arrivo di un lift per potersi dissolvere con esso, fungendo, insomma, da condizione di terminazione per le operazioni di marcatura. In tal modo, infatti, renderemmo addirittura impossibile l'eliminazione dei tagli esponenziali lì eventualmente presenti fino all'arrivo dei lift e quindi il sistema ED perderebbe la sua desiderata completezza rispetto ad EAL o LAL.

La migliore soluzione al primo problema è di raffinare le regole che riguardano le riduzioni di un lift con un nodo logico, facendo in modo che il lift sia assorbito ogniqualvolta esca dalla profondità alla quale è stato creato. Per distinguere i due diversi

comportamenti che un lift può assumere di fronte ad un nodo logico, è sufficiente aggiungere una condizione a margine, che è formalmente illustrata dalle figure 6.3 e 6.5.

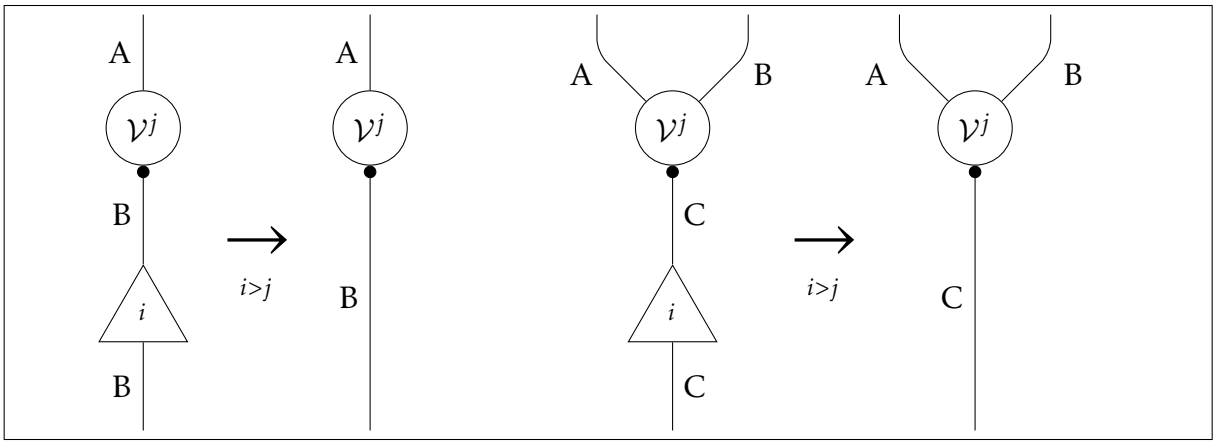


Figura 6.5.: Riscrittura a duplicazione esplicita: assorbimento di lift

Per risolvere anche la seconda questione, dobbiamo aumentare la consapevolezza dei nodi contrattori, che già gestiscono la nascita dei lift. Se ogni contrattore tiene traccia delle porte sulle quali è in attesa dell'arrivo di un lift, allora è in grado, allorquando effettui una successiva duplicazione, di evitare di generare un nuovo lift su tali porte, accollando il relativo lavoro al lift ritardatario. Del resto, si sa, chi tardi arriva *non* alloggia.

La figura 6.6 presenta la regola di propagazione del nodo contrattore (cfr. figura 6.2), che considera i casi del contrattore in attesa, i cui archi sono graficamente decorati con una punta, intende richiamare quella del lift e l'operazione di annichilamento.

6.1.4. Spazzatura

Sinora abbiamo assunto che se un lift entra in una scatola, cioè è di fronte alla sua porta principale, prima o poi ne uscirà, debitamente replicato da ogni propagazione attraverso i nodi ternari, da ogni porta secondaria della medesima scatola. Ma, con le regole fin qui informalmente introdotte, ciò non è necessariamente vero.

Se una scatola contiene infatti una sottorete spazzatura, che come premesse (estremi negativi) ha solo nodi (W) di indebolimento, e come conclusioni (estremi positivi) ha solo nodi ($L_!$) di esponenziazione, essa anzitutto non viene marcata dalla propagazione di lift, dacché questi ultimi sono chiaramente non raggiungibili a partire dalla porta principale. In tale scenario sarebbe, ancor peggio, pregiudicata ogni eventuale successiva marcatura, per le quale il contrattore in vana attesa evita di creare nuovi lift.

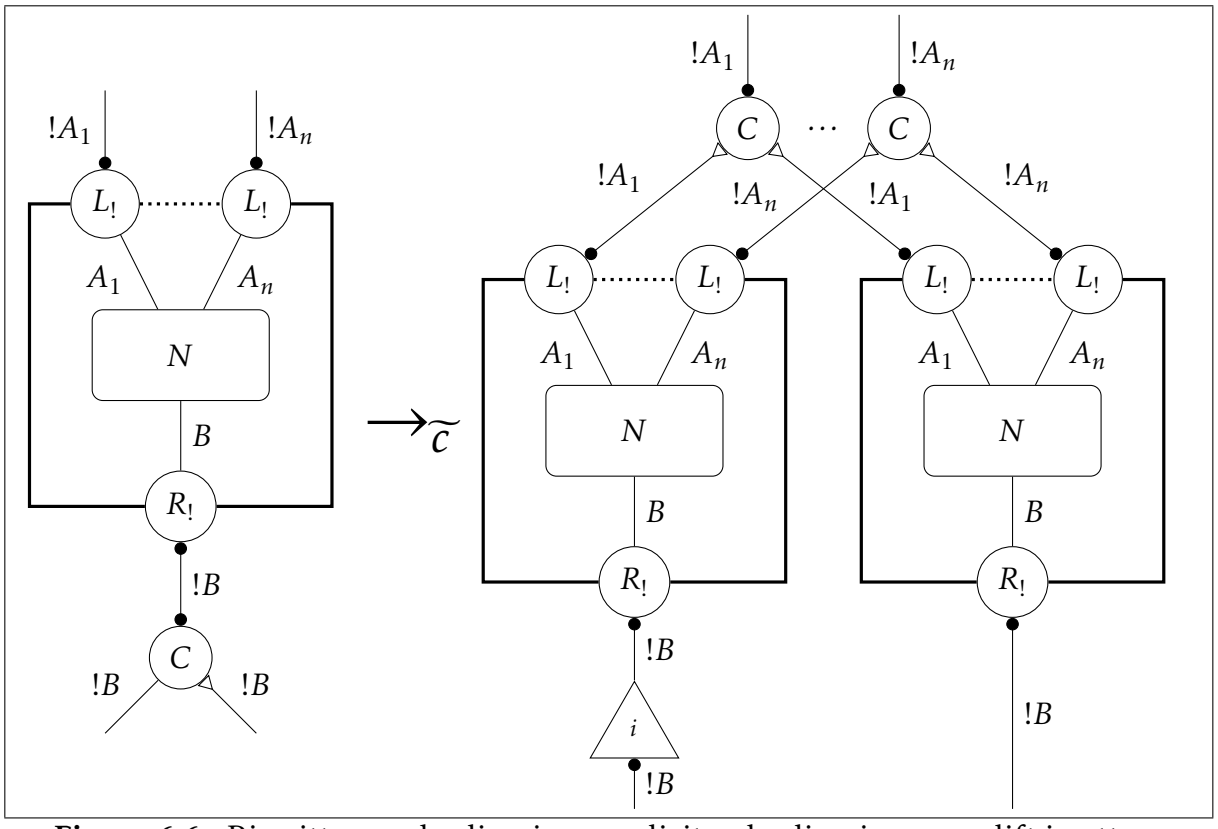


Figura 6.6.: Riscrittura a duplicazione esplicita: duplicazione, con lift in attesa

Ma come già accennato in 5.5, l'eliminazione di tali sottoreti ha una complessità al più lineare nella dimensione della rete che li contiene, quindi il primo dei due problemi risulta di entità trascurabile, pertanto, tranquillamente ignorabile. Ma lo stesso vale anche per il secondo, dato che i lift mancheranno di marcare esclusivamente quelle aree della rete di prova che sono collegate al nodo d'indebolimento, che insomma sono spazzatura.

Per questo motivo anche l'eliminazione del taglio a duplicazione esplicita ignora la presenza di spazzatura e le regole per il suo raccoglimento.

6.2. Riduzione a duplicazione esplicita

6.2.1. Definizione

Ora che appaiono chiari non solo gli obiettivi del sistema a duplicazione esplicita, ma anche le modalità di implementative, è possibile formulare compiutamente la riscrittura. Essa non modifica, ma estende, le regole di riscrittura della riduzione su logiche affini, al solo scopo di esplicitare in passi di riscrittura i costi della duplicazione. La definizione della relazione di riscrittura è pertanto formulata per differenza rispetto all'eliminazione del taglio delle logiche affini.

Definizione 6.1 (Eliminazione del taglio con duplicazione esplicita)**.** La relazione $\to_{ED}$ di eliminazione del taglio con duplicazione esplicita è uguale a $\to_E AL$, eccetto:

- eliminazione della regola di duplicazione, relativa al nodo (C)
- aggiunta delle regole di duplicazione esplicite, indicate con $\to_{\widetilde{c}}$ e rappresentate nelle figure 6.2 e 6.6,
- aggiunta delle regole di propagazione dei lift, indicate con $\to_\pi$, che realizzano avanzamento di lift (figg. 6.3 e 6.5) e incontro di lift (fig. 6.4).

Definizione 6.2 (Rete di prova a duplicazione esplicita)**.** Se N è una rete di prova EAL o LAL e $N \longrightarrow^*_{ED} N'$, allora N' è una rete di prova a duplicazione esplicita.

Si ricordi che la relazione $\to^*_{ED}$ è la chiusura riflessiva e transitiva della relazione $\to_{ED}$. Per distinguere le reti ED da quelle di EAL, segneremo queste ultime con una tilde, e.g. $\widetilde{N}$.

6.2.2. Correttezza e completezza

Requisito fondamentale per il sistema a duplicazione esplicita è il rapporto di estensione rispetto all'eliminazione del taglio delle logiche, quindi ci aspettiamo che esso esibisca correttezza e completezza, a meno delle differenze introdotte.

A tal fine, la seguente proposizione, dalla dimostrazione immediata, costituisce un primo risultato.

Definizione 6.3 (π-normalità)**.** Se una rete a duplicazione esplicita $\widetilde{N}$ non è riscrivibile per mezzo di $\rightarrow_\pi$, essa è detta in forma π-normale.

Proposizione 6.1. *Se una rete a duplicazione esplicita $\widetilde{N}$ è π-normale, allora è una rete EAL o LAL.*

La proprietà di correttezza rispetto ai sistemi EAL/LAL, ci assicura inoltre che ogni derivazione di ED ne sia immagine valida.

Proposizione 6.2. *(Correttezza rispetto EAL/LAL) Per ogni N, rete di prova di EAL/LAL, e per ogni riduzione $N \longrightarrow^*_{ED} \widetilde{N'}$, esiste una riduzione $N \longrightarrow^*_{EAL} N'$ tale che N' ed $\widetilde{N'}$ sono identiche, modulo eliminazione dei lift e ripulitura dei contrattori degli indici e degli indicatori d'attesa.*

Dimostrazione. La riduzione di EAL/LAL può essere banalmente costruita a partire da quella su ED, ignorando le regole dell'insieme π ed usando la regola di duplicazione originale al posto di $\widetilde{c}$ del sistema ED. □

Ancor più semplice è la dimostrazione di completezza di ED rispetto alle logiche di cui intende appunto essere un'estensione.

Proposizione 6.3. *(Completezza rispetto EAL/LAL) Per ogni rete di prova N e per ogni passo di riduzione $N \longrightarrow_{EAL} N'$, esiste una riduzione $N \longrightarrow^+_{ED} N'$, tale che N' è una rete di prova di EAL/LAL.*

Dimostrazione. Il passo può essere o una regola comune al sistema ED, e quindi la riduzione su di esso è trivialmente identica, o una duplicazione C, e allora esiste $N \longrightarrow_{\widetilde{c}} N'' \longrightarrow^*_\pi N'$ ed N' è la forma π-normale di N''. □

6.3. Complessità

Abbiamo appena visto come i costi di duplicazione dell'eliminazione del taglio delle rete di prova delle logiche leggere EAL o LAL possano essere tradotti in riduzioni di ED: in questa sezione vedremo che la lunghezza di tali riduzioni resta nelle classi di complessità rispettivamente elementare o polinomiale, in accordo con quanto stabilito dal teorema 5.1. Lungo la strada per tale meta, troveremo riscontro del comportamento della marcatura effettuata dai lift: essa è eseguita esattamente una volta su ogni nodo di ogni scatola copiata.

La completezza di ED rispetto a EAL/LAL (proposizione 6.3) ci offre un punto di partenza per aggredire il problema, visto che per ogni riduzione r di ED abbiamo una riduzione s su EAL/LAL tale che $|s| \leq |r|$. Ma in virtù dello stesso teorema 5.1 sappiamo che esiste una funzione elementare (o polinomiale) che limita la lunghezza di s e, per come è stato costruito ED. Pertanto, volendo individuare un limite per $|r|$, è sufficiente trovarne uno per il numero di quei passi aggiuntivi che r eventualmente

esegue rispetto ad s, vale a dire ai passi dell'insieme π, che operano la propagazione dei lift. È con quest'intento che procederemo in questa sezione: prima sarà introdotto qualche strumento tecnico, ma indispensabile alla dimostrazione; poi analizzeremo la complessità delle propagazioni, terminando infine stabilendo un limite alle derivazioni in generale.

6.3.1. Indicizzazione delle duplicazioni

Per analizzare nel dettaglio il comportamento dei lift, risulterà comodo poter fare riferimento alla specifica duplicazione per la quale stanno esplicitando il lavoro. Possiamo quindi supporre che i passi $\widetilde{c_i} \in r$ siano indicizzati progressivamente e che alla generazione di ogni lift questo sia decorato con il medesimo indice i. Ci riferiremo ad esso con più comodità chiamandolo lift-i, avendo cura di non confondere tale indice con la profondità di livello.

Come abbiamo visto nella precedente sezione 6.1.3, se un contrattore non ha ancora assorbito qualcuno dei due lift da esso creati in precedenza, lascerà che siano questi a esplicitare il successivo lavoro di duplicazione dello stesso contrattore, senza creare nuovi lift. Per gestire anche per questa situazione, introduciamo un nodo decorativo, detto lift fantasma, che cambia coerentemente l'indice del lift che lo attraversa. Se un contrattore è in attesa su una delle sue porte secondarie di un lift-i e opera la duplicazione j, allora lascia sull'arco che insiste su tale porta un lift fantasma $[j/i]$. Quando poi un lift-i raggiunge un fantasma $[j/i]$, diventa un lift-j. Nel caso in cui avvenga una riduzione che coinvolge un taglio sul quale insiste un lift fantasma, esso viene spostato su ogni nuovo collegamento creato nella rete.

Si noti che, benché le indicizzazioni dei lift e il comportamento dei fantasmi siano gestiti coerentemente al sistema di riscrittura ED, esse non hanno alcun significato computazionale, ma sono solo strumento dimostrativo.

6.3.2. Marcatura delle duplicazioni

I nodi lift hanno il compito di propagarsi lungo le sottoreti che sono state duplicate da passi di tipo $\widetilde{c}$, quindi ci aspettiamo che un limite superiore alla quantità di passi π dipenda dalla dimensione di tali sottoreti.

Risultando, quindi, utile discernere quali nodi siano stati attraversati dai lift di una certa duplicazione i, introduciamo una decorazione ausiliaria alla dimostrazione, lo stato di un nodo, che ci permetterà di provare un'invariante cruciale sul comportamento dei lift.

Definizione 6.4 (Stato dei nodi)**.** Consideriamo una riduzione $N \longrightarrow_{ED} N'$ e assumiamo che sia l'i-esimo passo di tipo $\widetilde{c}$ di una qualsivoglia riduzione multi-passo. Lo stato degli archi appartenenti a N' è:

- *marcato*, per i due archi posteriori ai due lift, propri o fantasma, introdotti in i,

- *non marcato*, per gli archi adiacenti ad ogni altro nodo appartenente a N e coinvolto dalla duplicazione i,
- *irrilevante*, per ogni altro arco.

Inoltre, quando viene eseguito un passo di propagazione di un lift, l'arco lungo il quale è fluito diventa *marcato*.

I lift-j con $j \neq i$ sono del tutto trasparenti all'assegnamento di stato relativo al passo di duplicazione i, e pertanto vengono ignorati. Ancor di più, diremo che un lift-i è di fronte ad un certo nodo anche quando fra i due è, in realtà, interposta una fila di arbitraria lunghezza di lift-j, con j comunque diverso da i. Questa assunzione non pregiudica in alcun modo la generalità dei risultati, dal momento che siamo certi che, prima o poi, ogni lift-j progredirà in un passo di propagazione, il cui costo non influenza i lift-i, visto che ai lift non sono consentiti sorpassi.

Per poterci riferire comodamente a quella porzione di una rete che ci aspettiamo sia marcata dai lift di una certa duplicazione, ne definiamo formalmente l'insieme dei nodi residui.

Definizione 6.5 (Nodi residui)**.** Sia M una rete di prova di ED e sia $M \longrightarrow_{\widetilde{c}} N$ un suo passo di duplicazione, che genera una sottorete $B \subseteq N$. L'insieme $B' \subseteq N'$ di nodi residui di i rispetto ad r, riduzione $N \longrightarrow^*_{ED} N'$ è definito induttivamente come segue.

Se r ha lunghezza nulla, allora $B' = B$.

Se al contrario r ha almeno un passo r_1, sia N'' quella rete e sia r_2 quella riduzione per le quali $N \xrightarrow{r_1}_{ED} N'' \xrightarrow{r_2}_{ED} N'$. Il residuo finale B' è uguale al residuo, rispetto alla riduzione r_2, di un certo $B'' \subseteq N''$ residuo di B rispetto al primo passo r_1. A seconda della natura di r_1, B'' è così composto:

- se r_1 non ha riguardato nodi di B, allora $B'' = B$;
- se r_1 ha cancellato un insieme di nodi $E \subseteq B$, allora $B'' = B \setminus E$;
- se r_1 è un passo $\widetilde{c}$ su una scatola $B_1 \supseteq B$, allora B viene duplicata insieme a B_1 in due immagini B ed B_c, e quindi $B'' = B \cup B_c$;

Attraverso lo stato degli archi della rete e gli insiemi residui, possiamo dimostrare che per un certo passo di duplicazione i, il comportamento dei lift-i è conforme alle nostre attese.

Proposizione 6.4 (Marcatura)**.** *Sia $N \longrightarrow^*_{ED} N'$ una riduzione di qualsivoglia lunghezza e sia $B' \in N'$ il residuo di un precedente passo di duplicazione, l'i-esimo. Allora*

(1) ogni nodo logico appartenente a B' residuo di $S \subseteq N'$, ha gli archi adiacenti con lo stesso stato, o marcato o non marcato (mai irrilevante);

(2) ogni lift-i, proprio o fantasma, ha arco posteriore con stato marcato e anteriore non marcato;

(3) i nodi con archi non marcati possono essere solo: nodi appartenenti a B', lift-i, nodi C in attesa di un lift-i, fantasmi [x/i] (ramo posteriore), o fantasmi [i/x] (ramo anteriore).

Dimostrazione. Per induzione sulla riduzione, è facile mostrare che ad ogni regola di riduzione preserva tali proprietà. □

La proposizione 6.4 ci mostra anzitutto come all'interno del grafo si possano sempre distinguere due categorie di archi rilevanti alle propagazioni dei lift-i. Infatti i lift-i, i lift fantasma e i nodi di attesa formano insieme una frontiera che separa i nodi logici di B' in due insiemi: quelli marcati, cioè raggiunti da un lift, e quelli non marcati, che ancora non lo sono stati. Il numero di nodi marcati sarà di seguito indicato con $\|B'\|$.

Una prima interessante conseguenza è che, come atteso, lo stato dei nodi può passare solo dallo stato marcato a quello non marcato. Inoltre, a partire dal fatto (1) della stessa proposizione e dalle regole di riduzione, possiamo estendere lo stato (marcato o non marcato) ad ogni nodo dell'insieme dei residui di B, a seconda dello stato dei nodi adiacenti.

6.3.3. Lunghezza delle propagazioni

Come accennato anche in precedenza, distinguiamo le regole di riduzione dell'insieme π in due tipologie: le propagazioni di avanzamento, quando il lift, di fronte ad un nodo binario (rispettivamente ternario), si propaga attraverso l'altro arco (gli altri due archi) dello stesso o svanisce; e quelle di incontro, quando il lift, di fronte ad un altro lift, si annichilisce o si scambia con esso.

Presa una derivazione r, indicheremo con $\widetilde{C}_r$ l'insieme delle duplicazioni, cioè riduzioni di tipo $\widetilde{c}$ appartenenti ad r, e con E_r l'insieme dei passi di eliminazione di nodi logici, che non siano cioè né propagazioni di lift π, né duplicazioni.

Per dare un limite superiore al numero di passi di avanzamento ci avvarremo della precedente proposizione che ci offre il modo di correlarli facilmente al numero di marcature.

Lemma 6.1 (Costo dell'avanzamento dei lift-i)**.** *Il numero dei passi di avanzamento dei lift-i in una riduzione $N \xrightarrow{r}_{ED} N'$ è limitato superiormente da*

$$\|N'\| + 2|E_r|.$$

Dimostrazione. Per induzione su r, è facile notare che:

- ogni passo in π e di avanzamento che coinvolge un lift-i marca esattamente un nodo in B, residuo della i-esima duplicazione,

- ogni passo in E_r cancella al più due nodi marcati in B,
- ogni passo in $\widetilde{c_r}$ crea solo nuovi nodi marcati,
- ogni altro passo di riduzione è ininfluente.

Quindi possiamo concludere che, al massimo, il numero dei passi di avanzamento è dato dalla somma fra il numero dei nodi marcati in N' e quelli che lo sono stati prima di essere cancellati insieme alle riduzioni in E_r, che quindi non appaiono più in N'. Nel peggiore dei casi tali eliminazioni hanno coinvolto solo nodi marcati e dunque il limite è verificato. □

Per le propagazioni di incontro, la questione si fa più complicata, costringendoci a considerare la presenza di lift di altri livelli, sinora ignorata.

Lemma 6.2 (Costo degli incontri di lift)**.** *Presa una riduzione $N \xrightarrow{r}_{ED} N'$, il numero di passi di incontro fra lift-i e lift-j (con i eventualmente uguale a j), è superiormente limitato da*

$$2\left(\|N'\| + 3|E_r|\right)\left(|\widetilde{C}_r| + 1\right)^{\partial(N)}.$$

Dimostrazione. Mostriamo per induzione sul livello l e sulla derivazione r che per ogni scatola di livello l appartenente a N', ci sono un numero di incontri di lift, (annichilamenti o scambi), creati dalla riduzione r che è al più

$$2\left(\|N'\| + 3|E_r|\right)\left(|\widetilde{C}_r| + 1\right)^{\partial(N)-l}$$

Caso base: $l = \partial(N)$. Al livello di profondità massimo, il numero di incontri di lift può aumentare per mezzo di un passo $\widetilde{c}$ solo qualora esso crei una nuova scatola con lo stesso numero di incontri, giacché non possono essere creati nuovi incontri all'interno di una scatola di livello l. Per gli altri casi, analogamente a quanto abbiamo visto in precedenza, possiamo notare che

- ogni passo di avanzamento di lift crea al più due nuovi incontri e marca un nodo,
- ogni regola in E_r crea al più due incontri e cancella al più due nodi marcati,
- ogni altro passo di riduzione non crea nuovi incontri.

Quindi il numero di incontri di lift creati al livello $\partial(N)$ è al più:

$$2\|N'\| + 6|E_r|$$

e la tesi pertanto è verificata. ◇

Caso induttivo: $l < \partial(N)$. Per qualunque altro livello l, continuano ad essere valide le considerazioni precedenti, tranne che per i passi di duplicazione, che al peggio sono disposti in una concatenazione che dà luogo ad un esponenziazione di parametro pari alla distanza dalla profondità massima, ovvero $\partial(N) - l - 1$. In questo caso, pertanto, il numero di incontri creati è al più

$$(2\|N'\| + 6|E_r|)\,|\widetilde{C}_r|^{\partial(N)-l-1},$$

e questo dimostra il caso induttivo, ◇

e conclude la prova. □

6.3.4. Lunghezza delle riduzioni

Il risultato finale di questa sezione ci consente infine di stabilire un limite superiore alla lunghezza delle riduzioni ED a partire dalla quantità di nodi logici marcati nel grafo ridotto.

Teorema 6.1 (Complessità di ED)**.** *Per ogni naturale d esiste una funzione elementare (rispettivamente polinomiale) f_d tale che per ogni rete di prova N in EAL (LAL) per la quale $\partial(N) = d$, se $N \xrightarrow{r}_{ED} N'$, allora $|r| \leq f_d(|N|)$.*

Dimostrazione. Il numero di passi di duplicazione, il numero di nodi marcati e le riduzioni dell'insieme E_r sono facilmente limitabili sfruttando la correttezza di ED rispetto ad EAL e LAL (proposizione 6.2) e il limite superiore della lunghezza delle riduzioni EAL (teorema 5.1).

Ma il numero di propagazioni di lift, avanzamenti e incontri, relativi ad una duplicazione è stato ben limitato proprio in funzione del numero di nodi marcati in N e dei passi di contrazione, rispettivamente dai lemmi 6.1 e 6.2. Quindi è possibile dedurre l'esistenza della funzione f_d. □

7. Implementazione condivisa

> Tutte le verità sono facili da capire
> una volta che siano state rivelate;
> il punto è rivelarle.
>
> Galileo Galilei (1564-1642)

Stabilito il sistema di riscrittura a duplicazione esplicita, in questo capitolo concluderemo l'analisi della complessità dell'implementazione condivisa nel caso di λ termini tipabili in EAL o LAL. Dopo aver definito formalmente l'implementazione di reti di prova in grafi di condivisione mostreremo come la riduzione condivisa sia simulabile attraverso la riduzione a duplicazione esplicita. Proprio tale risultato consentirà, nella sezione conclusiva del capitolo, di dimostrare in maniera semplice la correttezza dell'implementazione e l'atteso limite di complessità nella lunghezza della riduzione condivisa: essa resta nelle classi di complessità proprie delle logiche affini di riferimento.

7.1. Implementazione condivisa di reti EAL/LAL

L'implementazione condivisa è definita da una funzione di traduzione iniziale e dal sistema di riduzione condivisa.

7.1.1. Traduzione iniziale

Dai termini e dai tipi di EAL/LAL, risulta già evidente che il costruttore di tipi funzionale corrisponde all'implicazione lineare $\multimap$, e che pertanto la traduzione iniziale da reti di prova a grafo di condivisione non potrà che mappare l'implicazione sinistra nell'applicazione e quella destra nell'astrazione. Inoltre i tipi $\forall$ e μ non si manifestano nel livello dei λ termini, quindi non appariranno nemmeno nei grafi della riduzione condivisa.

Per quanto riguarda le scatole, ricordiamo anzitutto (cfr. sez. 5.2.2) che i tipi §, ferme restanti le restrizioni sulla formazione delle rete di prova di LAL e quindi le relative regole di tipizzazione, sono trattabili esattamente come quelle di tipo !.

In generale, nel mondo dei grafi di condivisione, la necessità di scatole e di tipi esponenziali che vincolano la duplicazione e ne delimitano lo *scope*, cessa totalmente. Infatti, tali vincoli non sono presenti nella sintassi dei termini ed i confini sono

realizzati dai fan, che gestiscono anche la contrazione (staticamente) e la duplicazione (dinamicamente), deputate ai nodi (c) delle reti di prova.

L'ultimo dettaglio è l'indicizzazione per i fan, che si è scelta essere assegnata in base alla profondità, o livello, del nodo (vedi def. 5.10).

Definizione 7.1 (Traduzione EAL/LAL)**.** La funzione di traduzione $\mathcal{T}$ da reti di prova di EAL in grafi di condivisione è definita come segue:

- $\mathcal{T}(R_{\multimap}) = \lambda$,
- $\mathcal{T}(L_{\multimap}) = @$,
- $\mathcal{T}(C) =$ fan, con indice $\partial(C)$,
- nodi in $\{L_!, R_!, L_\forall, R_\forall, L_\mu, R_\mu\}$ eliminati,
- scatole eliminate.

7.1.2. Riduzione condivisa

Invece di considerare il completo sistema di riscrittura su grafo di condivisione possiamo comodamente eliminare certe inutili porzioni per rendere minimale la trattazione.

Oracolo

La riduzione condivisa non fa uso di scatole o modalità, che la obbligherebbero ad effettuare duplicazioni globali, cioè una scatola alla volta, ma piuttosto mantiene indici di profondità per ognuno dei collegamenti, provvedendo al loro aggiornamento lungo la riduzione. L'operazione di manutenzione degli indici, gergalmente chiamata la contabilità dei livelli (dall'inglese *bookkeeping*), può essere implementata per mezzo di nodi aggiuntivi, il *bracket* ed il *croissant*, che, rispettivamente, alzano e abbassano temporaneamente il livello di una sottorete e la cui propagazione ha il ruolo di reindicizzare altri vertici. Una soluzione più sintetica consiste nell'affidare la gestione direttamente ai nodi di condivisione e duplicazione, che in questo caso sono detti *mux*.

Ma la proprietà di stratificazione per le reti di prova dei sistemi EAL e LAL (vedi 5.2) ci assicura che il livello di un vertice è invariante rispetto all'eliminazione del taglio. Quindi anche i nodi dei grafi condivisi sono stratificati e quindi l'oracolo, che implementa la contabilità dei livelli, non è affatto necessario e possiamo limitarci all'algoritmo di riduzione astratto.

Questo rende lo studio della complessità dell'implementazione condivisa più facilmente affrontabile e spiega come mai la sua analisi sia stata inizialmente approcciata in questo contesto semplificato.

Spazzatura

I grafi di condivisione possiedono, come abbiamo visto nella sezione 3.1.1, un insieme di regole di riscrittura che eliminano la spazzatura, ovvero quei termini non legati e quindi tipati per mezzo di regole di indebolimento.

Avendo deciso in 5.5, per semplicità di trattazione, di tralasciare le regole di eliminazione della spazzatura dalle reti di prova EAL, le corrispettive regole di riscrittura per i grafi di condivisione risultano inutili e, pertanto, ignorate.

Riduzione su grafi di condivisione

Pertanto, le regole di riscrittura su grafi di condivisione rappresentate nelle seguenti figure e che qui consideriamo, sono un un sottoinsieme di quelle presentate nel capitolo 3.

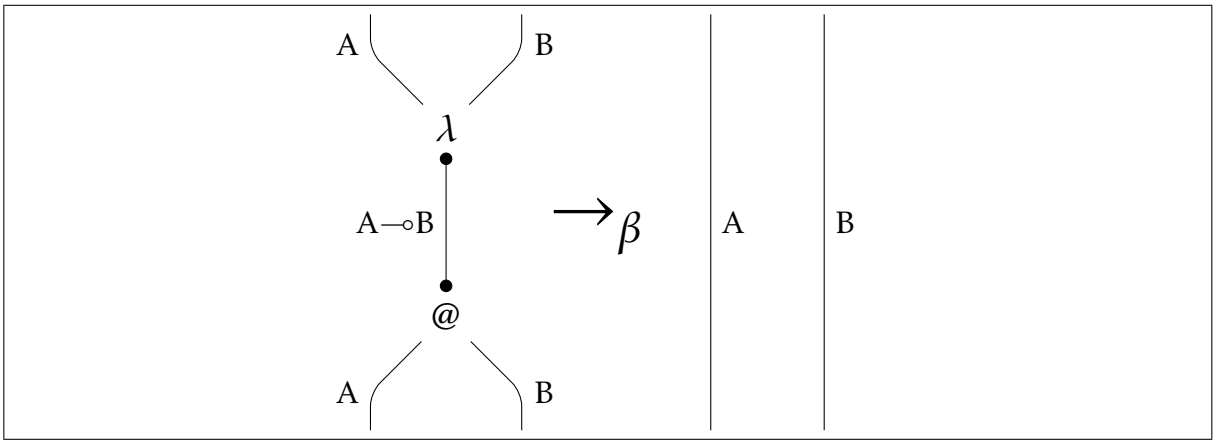


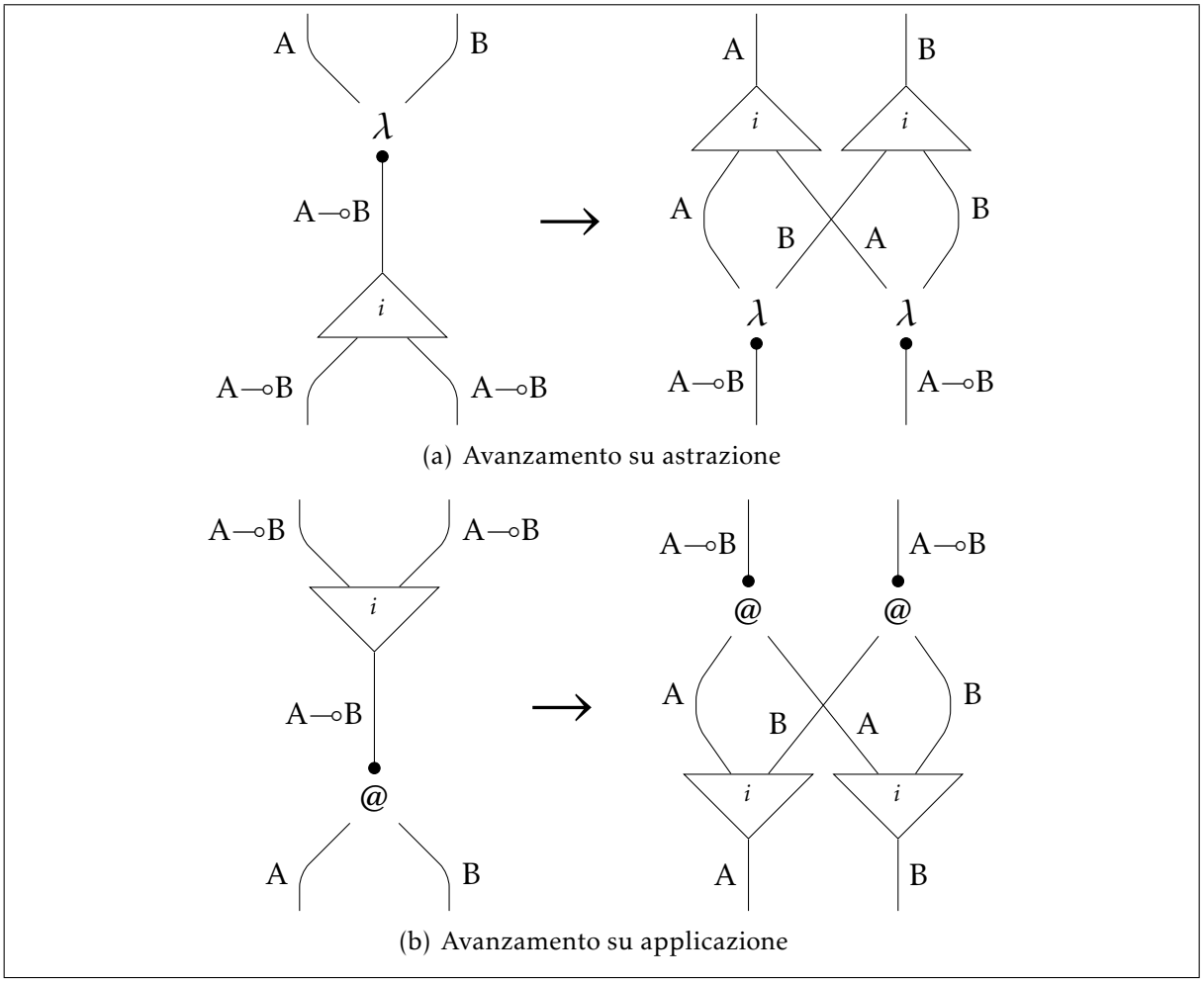
Figura 7.1.: Riduzione condivisa: β riduzione

Le regole di riscrittura generano il sistema nel seguito chiamato SG.

Definizione 7.2 (Riduzione condivisa)**.** La relazione $\longrightarrow_{SG}$ di riduzione condivisa, è la minima relazione indotta dalle seguenti regole di riscrittura:

- β riduzione(fig. 7.1),
- scambio di fan (fig. 7.3, sinistra),
- annichilamento di fan (fig. 7.3, destra),
- avanzamento attraverso astrazione (fig. 7.2(a)),
- avanzamento attraverso applicazione (fig. 7.2(b)).

Si ricordi che per rendere la riduzione ottimale secondo Lévy, sarebbe necessario imporre l'applicazione pigra, ovvero con chiamata per necessità, delle regole di propagazione (cfr. sez. 3.4), ma essendo interessati al costo intrinseco della riduzione condivisa, ci occuperemo di ogni strategia, inclusa quella ottimale.



(a) Avanzamento su astrazione

(b) Avanzamento su applicazione

Figura 7.2.: Riduzione condivisa: avanzamento di fan

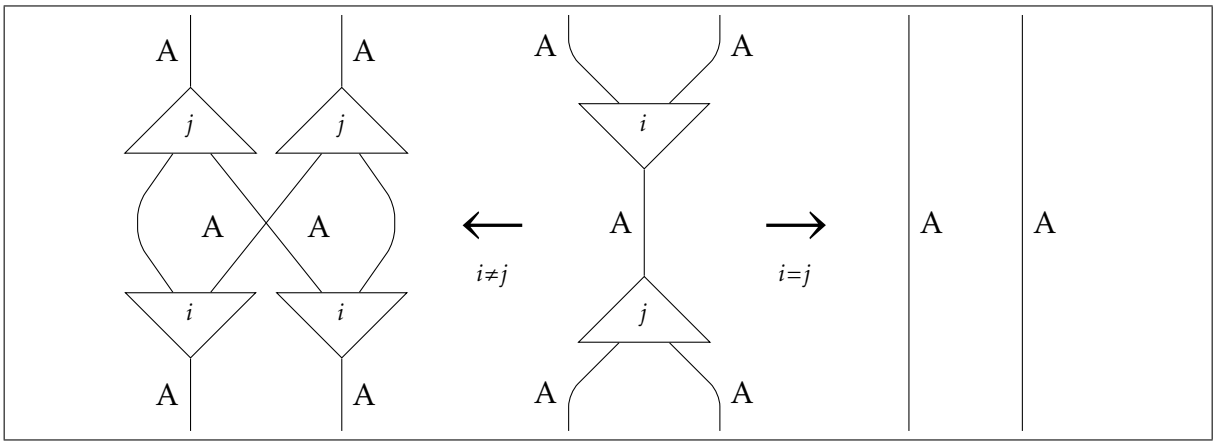
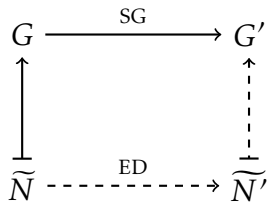


Figura 7.3.: Riduzione condivisa: incontro fra fan, scambio o annichilamento

7.2. Simulazione

Le riduzioni condivise sono simulabili per mezzo del sistema a duplicazione esplicita ovvero ogni passo di riduzione condivisa ha una controparte sulla corrispondente rete di prova. Individuata l'adeguata strategia di riduzione su ED e l'intesa relazione fra reti di prova ED, è possibile stabilire una simulazione come raffigurato dal diagramma:



7.2.1. Equivalenza di copia

Per cominciare abbiamo bisogno di definire una relazione fra quei nodi e quei lift che sono stati originati da una medesima operazione di duplicazione.

Definizione 7.3 (Equivalenza di copia)**.** Sia $\widehat{N} \longrightarrow^{+}_{ED} \widehat{N'}$ una riduzione che contiene un passo di duplicazione i. Due vertici v' ed v'' in $\widehat{N'}$ sono in *equivalenza di copia* per i se sono generati da i e

- v' e v'' sono due lift, oppure
- v' e v'' sono due nodi generati da un medesimo nodo $v \in \widehat{N}$.

La relazione è chiaramente riflessiva, simmetrica e transitiva, e questo giustifica il nome e la notazione $\equiv_i$, dove i è l'indice della duplicazione comune. La classe d'equivalenza per un vertice v rispetto alla duplicazione i è invece indicata, come consuetudine, con $[v]_{\equiv_i}$

7.2.2. Riduzione ED sincrona

Ora siamo in grado di definire una strategia di riduzione che renda la dinamica delle reti a duplicazione esplicita più facilmente conformabile a quella dei grafo di condivisione, imponendo sincronia per le regioni equivalenti rispetto alla copia e non marcate (vedi sez. 6.3.2) e per i lift.

Definizione 7.4 (Riduzione ED sincrona)**.** Una riduzione r per cui $\widehat{N} \longrightarrow^{*}_{ED} \widehat{N'}$ è detta sincrona se verifica una delle seguenti condizioni:

1. r è vuota, ovvero $\widehat{N} = \widehat{N'}$,

2. r è il minimo insieme $r_1, \ldots, r_n$ di passi di propagazione, ognuno su un elemento diverso della classe $[l]_{\equiv_i}$ per un lift $l \in \widetilde{N}$,

3. r è il minimo insieme $r_1, \ldots, r_n$ di passi di eliminazione di tagli, ognuno su un elemento diverso della classe $[n]_{\equiv_i}$ per un nodo $n \in \widetilde{N}$ con stato marcato.

La seconda e la terza condizione impongono che, fissata una duplicazione i-esima, le classi d'equivalenza di copia per lift e nodi marcati siano ridotte necessariamente all'unisono: se si effettua un passo relativo ad uno di questi insiemi, un'identica riscrittura è applicata all'intero insieme.

Si congettura che tale definizione, limitatamente ai nodi non marcati, sia una formulazione equivalente e più sintetica della riduzione completa di una famiglia di redex (cfr 2.3).

Chiameremo sincronizzate quelle reti ED che sono raggiungibili con strategia sincrona a partire da una rete di prova di EAL. Tali reti saranno proprio l'immagine della simulazione della riduzione su grafi di condivisione.

Definizione 7.5 (Rete ED sincronizzata)**.** Una rete ED $\widetilde{N}$ è detta *sincronizzata* se verifica una delle seguenti condizioni:

1. $\widetilde{N}$ è una rete di prova di EAL, oppure

2. esiste una rete ED sincronizzata $\widetilde{M}$ tale che $\widetilde{M} \longrightarrow_{ED}^{*} \widetilde{N}$ e la riduzione è sincrona.

In altre parole, le reti sincronizzate sono la più piccola classe contenente le reti EAL e chiusa rispetto alla relazione di riduzione sincrona. Ci riferiremo a reti di questo tipo apponendo indicando lettere maiuscole segnate con una barra orizzontale, e.g. $\overline{N}$.

7.2.3. Relazione ED-SG

L'ultimo ingrediente che manca per poter definire la simulazione, è una relazione fra reti a duplicazione esplicita e grafi di condivisione. La diversità fra lo stile di duplicazione dei sistemi di riscrittura EAL ed SG, per l'una quantizzato, per l'altra incrementale, passa inosservata quando l'implementazione condivisa funge da mero strumento di normalizzazione. Ma nel nostro caso vogliamo evidenziare proprio tale differenza, rompendo la canonica corrispondenza fra nodi contrattori di EAL e fan di SG, stabilita per la traduzione iniziale $\mathcal{T}$.

Quando una rete ED è π-normale e il lavoro di duplicazione "pagato", ogni fan può corrispondere ad un contrattore, ma quando ci sono lift ancora in azione vogliamo che le regioni ancora non marcate siano fatte corrispondere a nodi non ancora duplicati del grafo di condivisione. Formulata come una funzione fra reti ED e grafi SG, ci aspettiamo quindi che essa sia suriettiva per ogni nodo della rete ed iniettiva rispetto all'equivalenza di copia per ogni duplicazione.

Estendiamo la traduzione iniziale $\mathcal{T}$ affinché soddisfi anche quest'ultima proprietà.

Definizione 7.6 (Traduzione da reti ED sincronizzate)**.** $\overline{\mathcal{T}}$ è una *funzione di traduzione* che mappa reti ED sincronizzate in grafi di condivisione tale che:

- i nodi di tipo L_μ, R_μ, $L_\forall$, $R_\forall$ sono cancellati,
- i nodi di tipo $L_!$, $R_!$ e le scatole sono cancellati,
- i nodi di tipo $L_{\multimap}$ e $R_{\multimap}$ sono mappati rispettivamente in nodi λ e @
- i nodi di contrazione con nodi in attesa sono cancellati,
- i nodi di contrazione sono mappati in un fan,

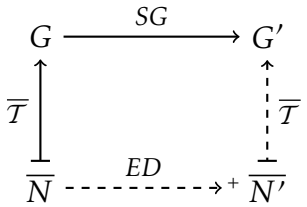
e che è iniettiva rispetto all'equivalenza di copia per ogni duplicazione:

- per ogni duplicazione i e per ogni coppia di nodi $n_1, n_2 \in \overline{N}$, vale che $n_1 \equiv_i n_2$ se e solo $\overline{\mathcal{T}}(n_1) = \overline{\mathcal{T}}(n_2)$.

7.2.4. Simulazione con ED

Siamo ora pronti per stabilire un interessante risultato sulla relazione fra la riduzione condivisa e l'eliminazione del taglio a duplicazione esplicita: la seconda è in grado di simulare la prima, modulo la relazione $\overline{\mathcal{T}}$. Preso, cioè, un qualunque grafo di condivisione che sia una traduzione condivisa di una rete di prova sincrona, ogni passo sul grafo ha una corrispondente riduzione a duplicazione esplicita sulla rete di prova, di lunghezza non nulla, che preserva la traduzione $\overline{\mathcal{T}}$.

Teorema 7.1 (Simulazione)**.** *Sia $\overline{N}$ una rete di prova sincronizzata di EAL e sia G il grafo di condivisione per cui $\overline{\mathcal{T}}(\overline{N}) = G$. Se $G \longrightarrow_{SG} G'$, allora esiste $\overline{N'}$ tale che $\overline{N} \longrightarrow^+_{ED} \overline{N'}$ e $\overline{\mathcal{T}}(\overline{N'}) = G'$. Ovvero:*



La sincronia delle riduzioni ED (definizione 7.4) gioca qui la sua parte, dato che è una condizione necessaria a garantirne la corrispondenza con la riduzione condivisa. Il motivo di tale necessità, se non già chiaro all'intuizione, è spiegato nella dimostrazione del teorema.

Dimostrazione del teorema di simulazione (7.1). Mostreremo l'esistenza delle simulazione costruendo una riduzione r in due fasi: prima (passo 1) effettuando una riduzione preliminare e', che esegue l'eventuale riduzione su aree della rete $\overline{N}$ che non hanno alcuna immagine nella rappresentazione a grafo di condivisione poi (passo 2) eseguendo una riduzione immagine e'', di lunghezza positiva, che invece è esattamente mappato dalla riduzione su G.

Siano g_1 e g_2 i due nodi interessati dalla riduzione e siano $P = \{p_1, \ldots, p_k,\}$ e $Q = \{q_1, \ldots, q_l,\}$ quei due sottoinsiemi di $\overline{N}$ immagine di g_1 e g_2 (cioè tali che per ogni $i < k$ e per ogni $j < l$, vale $\overline{T}(p_i) = g_1$ e $\overline{T}(q_j) = g_2$).

Passo 1 (Riduzione preliminare)**.** *Dato che la traduzione collassa solo nodi nella stessa classe d'equivalenza per una certa duplicazione i (cioè se $\overline{T}(n_1) = \overline{T}(n_2)$ allora esiste i per cui $n_1 \equiv_i n_2$), per ogni $p_i \in P$ c'è un unico $q_j \in Q$ che sia ad esso collegato tramite percorsi diretti (def. 5.6).*

Ma, per come è definita la funzione di traduzione, ogni percorso diretto (eventualmente lineare o addirittura unitario) può contenere solo nodi non mappati in G. Infatti se, per assurdo, esistesse un nodo $n \in \overline{N}$ mappato in un nodo $g_3 \in G$, allora esso sarebbe interposto fra g_1 e g_2, rendendo impossibile il relativo passo di riduzione ipotizzato.

Gli unici tipi di nodi che non hanno un'immagine in G sono μ, $\forall$, ! e $\overline{c}$ in attesa: i primi tre sono nodi binari (un ramo positivo ed uno negativo), l'ultimo è invece ternario (due rami negativi ed uno positivo). Pertanto i percorsi sono necessariamente strutturati come una foresta di l alberi con radici in Q e k foglie in P.

Inoltre, visto che i tipi agli estremi combaciano, esiste una riduzione, sincrona ed eventualmente nulla, che districa sincronicamente ogni diramazione, propagando ogni contrattore in attesa, e accorcia sincronicamente ogni ramo che è sequenza di tagli μ, $\forall$ e !. Tale riduzione, preservante i tipi e trasparente alla traduzione $\overline{T}$, aumenta l'insieme Q che è mappato in g_2, fino a quando la sua cardinalità non raggiunge k, rendendo ogni p_i direttamente collegato a q_j.

Passo 2 (Riduzione immagine)**.** *Le coppie di nodi su (p_i, q_j), ora sono sicuramente e direttamente collegate ed esiste una (unica) regola di riduzione applicabile ad ognuna di esse. Per come è definita $\overline{T}$ andiamo per casi sul tipo di riduzione su G individuandone le possibili corrispondenze:*

- *riduzione β: ogni coppia è del tipo $L_{\_\circ}$ e $R_{\_\circ}$*
- *avanzamento di un fan su λ (o @): lift e $R_{\_\circ}$ (o $L_{\_\circ}$))*
- *incontri di fan: lift e lift*

Le corrispondenze permettono sempre di eseguire un passo di riduzione su di ognuna delle coppie, mantenendo coerente la relazione di traduzione.

Passo 3 (Riduzione di simulazione)**.** *La concatenazione dell'eventuale riduzione preliminare e' e della riduzione immagine e'' è una riduzione valida su ED di lunghezza non nulla che simula il passo su SG.*

□

È interessante notare come, nella dimostrazione, le duplicazioni su ED non abbiano effettiva controparte nella simulazione di una riduzione SG: dato che la corrispondenza dei fan è mantenuta con i lift, nella costruzione tali duplicazioni rientrano nella riduzione preliminare.

7.3. Correttezza

Come già ricordato nella sezione 4.2, Baillot *ed altri* (2011) già assicurano correttezza e completezza dell'implementazione condivisa per gli stessi sistemi EAL e LAL, definendo una funzione di *readback* realizzata con la semantica dei contesti.

Ma possiamo notare, proprio come prima e semplice conseguenza del teorema di simulazione, che ogni riduzione condivisa, compresa quindi quella ottimale, è corretta rispetto ad una riduzione ED.

Corollario 7.1 (Correttezza per EAL)**.** *Sia N una rete di prova di EAL e sia G il grafo di condivisione per cui $\mathcal{T}(N) = G$. Se $G \longrightarrow^*_{SG} G'$, allora esiste N' tale che $N \longrightarrow^*_{EAL} N'$ e $\mathcal{T}(N') = G'$.*

Dimostrazione. Segue direttamente dall'esistenza di una simulazione ED di ogni passo di riduzione SG (teorema 7.1) e dalla correttezza di riduzioni ED rispetto ad EAL (proposizione 6.2). □

7.4. Complessità

Il teorema consente inoltre di dare un primo sguardo ai benefici dell'implementazione condivisa: le riduzioni SG sono non più lunghe delle corrispondenti ED, che esprimono in maniera ragionevole i costi di duplicazione.

Corollario 7.2 (Ottimalità per ED)**.** *Sia $\overline{N}$ una rete di prova sincronizzata di EAL e sia G il grafo di condivisione per cui $\overline{\mathcal{T}}(\overline{N}) = G$. Se $G \xrightarrow{s}_{SG} G'$, sia $\overline{N} \xrightarrow{r}_{ED} \overline{N'}$ la corrispondente simulazione. Allora $|s| \leq |r|$.*

Dimostrazione. Ovvio dal teorema 7.1 di simulazione. □

La differenza di lunghezza proviene dal fatto che ogni passo su SG corrisponde ad una riduzione sincrona non nulla su ED. Se anche contassimo come unitarie le azioni

di propagazione di lift, cioè avanzamenti ed incontri, presenti nella riduzione preliminare con la quale abbiamo costruito la riduzione r su ED, e se, ancora, ignorassimo ogni eliminazione di tagli relativi a nodi logici non mappati nell'implementazione condivisa, avremmo una riduzione comunque più che unitaria. Infatti, i nodi non marcati in N sono collassati in G e quindi in quei casi l'immagine r è necessariamente più lunga di s. È proprio in questo disavanzo di lunghezza che si manifesta uno dei caratteri d'ottimalità dell'implementazione condivisa: la duplicazione è un passo minimale.

7.4.1. Lunghezza delle riduzioni

Il più importante risultato di complessità per SG, è la limitazione della lunghezza delle riduzioni condivise a funzioni elementari o polinomiali, a seconda della logica affine di riferimento, elementare o leggera. Esso è stato pubblicato per la prima volta ad opera di Baillot *ed altri* (2007).

Teorema 7.2 (Complessità di SG)**.** *Per ogni naturale d, esiste una funzione elementare (rispettivamente polinomiale) f_d tale che, per ogni rete rete di prova N di EAL (LAL), con $\partial(N) = d$, e per la quale $N \xrightarrow{r}_{EAL} N'$; esistono un grafo di condivisione G tale che $\mathcal{T}(N) = G$ e una riduzione condivisa $G \xrightarrow{s}_{SG} G'$ per la quale $|s| \leq f_{\partial(N)}(|r|)$.*

Grazie al precedente corollario, e quindi alla simulazione col sistema ED, il risultato può essere dimostrato in maniera originale.

Dimostrazione. Immediata, per riduzione, dato che la lunghezza della riduzione condivisa s è non superiore a quella di una riduzione su ED (corollario 7.2), per la quale esiste una funzione di classe di complessità adeguata che ne limita la lunghezza (teorema 6.1). □

Nel caso di sistemi stratificati, nei quali l'oracolo non complica l'analisi di complessità, è stata quindi ri-dimostrata l'adeguatezza della complessità dell'implementazione condivisa.

Parte III.

Conclusioni

8. Contributi

I contributi originali presentati in questa Tesi di Laurea Magistrale sono stati esposti lungo la trattazione della parte seconda, nei capitoli 6 ed 7 e sono riassunti nel presente.

Riduzione a duplicazione esplicita

Il sistema di riscrittura a duplicazione esplicita per reti di prova EAL o LAL, illustrato nel capitolo 6, è stato ideato come semplice strumento dimostrativo, ma risulta essere interessante anche altrimenti. Una sua generalizzazione renderebbe lo strumento riutilizzabile come ponte anche in altre situazioni, laddove si richieda un'analisi di complessità di altre implementazioni condivise o, ancor più in generale, fra due sistemi di riscrittura su grafi con diversi stili di duplicazione, locale e globale.

I marcatori lift che il sistema introduce permettono, infatti, di mantenere facilmente la corrispondenza con i nodi fan e, al contempo, di esplicitare i costi di duplicazione in passi di riscrittura, rendendo l'analisi di complessità elegantemente dimostrabile tramite simulazione.

Dimostrazione di correttezza

Delle dimostrazioni di correttezza di implementazioni condivise, si è trattato, nella sezione 4.2, citando la dimostrazione di correttezza debole nel caso di EAL/LAL presentata da Baillot *ed altri* (2011), effettuata per mezzo della semantica dei contesti.

Un risultato di complessità più forte, il corollario 7.1, è stato ottenuto indipendentemente per via sintattica, attraverso la simulazione stabilita nella sezione 7.2 della quale costiusce conseguenza immediata.

Dimostrazione di complessità

Il risultato sulla complessità dell'implementazione condivisa, sempre nel caso delle logiche EAL e LAL, ovvero la limitazione superiore sulla taglia della rete di prova è stato presentato originalmente da Baillot *ed altri* (2011). Il teorema 4.4 ha enunciato il risultato e la sezione 4.3.3 ha illustrato, per sommi capi, l'idea alla base della loro dimostrazione, condotta per via semantica.

Il medesimo risultato, formulato nel teorema 7.2, è stato alternativamente dimostrato per via diretta, senza necessitare della semantica dei contesti per le reti di prova, ma esplorando invece le corrispondenze sintattiche. Viene, in effetti, stabilita una simulazione della riduzione condivisa su ED, un sistema di riscrittura intermedio fra l'eliminazione del taglio per le reti di prova e la riduzione su grafi di condivisione. Per ogni passo di riduzione condivisa la simulazione corrispondente richiede una sequenza non nulla di riscritture su ED, la cui lunghezza, in generale, è dimostrata essere limitata secondo i risultati classici sulle logiche EAL e LAL.

9. Sviluppi

9.1. Sviluppi attuali

In collaborazione col correlatore Guerrini, il candidato autore della tesi, ha già intrapreso percorsi di lavoro che mirano ad estere i risultati presentati nella stessa e che sono attualmente in corso d'opera.

Accuratezza della limitazione dei costi

L'analisi di complessità dell'implementazione condivisa per termini tipati in EAL o LAL non è precisa, dato che la limitazione inferiore della complessità non è strettamente calcolata.

Una semplice estensione del risultato potrebbe computare precisamente il costo della riduzione condivisa, relazionandolo direttamente al costo di ogni riduzione, mostrando con più forza il risultato di ottimalità. A tal fine il sistema a duplicazione esplicita potrebbe essere accuratamente modificato affinché la propagazione dei lift abbia un costo precisamente corrispondente alle operazioni di duplicazione effettuate dalle riduzioni classiche.

Costi della riletura

Il sistema di rilettura dei grafi è implementabile direttamente per via sintattica, con regole di propagazione dei fan che possano interagire anche lungo archi secondari di altri nodi.

La dimostrazione qui presentata potrebbe essere estesa, includendo queste operazioni, provvedendo così a rendere completare l'analisi della complessità di questo caso dell'implementazione condivisa e facilmente riutilizzabile.

Lambda-calcoli leggeri

Le stratificazione del calcolo di riferimento si manifesta solo nelle logiche EAL e LAL, che costituiscono i sistemi di tipo.

Considerando dei calcoli con sintassi esplicitamente ristretta, come il λ calcolo affine leggero presentato da Terui (2007), potrebbe essere possibile disporre di ulteriori strumenti per l'analisi sintattica dell'ottimalità.

9.2. Sviluppi futuri

Modelli di costo per il λ calcolo

La riduzione condivisa offre ricchi spunti sulle dinamiche atomiche del λ calcolo e della sua riduzione: la legatura, la sostituzione, la copia, la gestione dei livelli. Ciononostante ancora manca un modello di costo per il lambda calcolo, una caratterizzazione, cioè, di una qualche misura sui termini che sia polinomialmente legata al costo necessario alla loro riduzione su una macchina Turing-equivalente.

Le famiglie di redex non sono la risposta, ma si intuisce che possano contenere suggerimenti significativi, che ancora non sono stati sviluppati.

Complessità nel caso generale

Il più importante problema aperte è quello di mostrare una limitazione superiore al costo della riduzione ottimale nel caso generale. Gli unici progressi in questo senso, sono nel già citato articolo di Baillot *ed altri* (2011) e, in misura più modesta, in questa tesi, ma ambedue i lavori si sono però dedicati all'algoritmo astratto il cui comportamento è più facilmente analizzabile.

Nonostante tutti i pochi esperti del settore ne congetturino la validità o addirittura, lo considerino *folklore*, manca ancora una dimostrazione dell'ottimalità dell'ottimalità.

10. Riconoscenze e riconoscimenti

Le prime riconoscenze sono per i relatori di questa tesi, che hanno accettato di indirizzare il mio lavoro a Bologna e a Parigi, durante i mesi di tirocinio al *Laboratoire d'Informatique de Paris-Nord* (LIPN), presso l'*Université Paris 13*. Ringrazio Simone Martini, che da docente ha sostanzialmente contribuito alla nascita della mia passione per la ricerca, poi, in qualità di relatore, mi ha sapientemente introdotto alla logica lineare e ai grafi di condivisione e ha puntualmente contribuito alla revisione della tesi. E ringrazio Stefano Guerrini, che mi ha seguito durante i periodi centrali del lavoro, stimolandomi con vivace acutezza e che ha, direttamente e indirettamente, ispirato molte delle idee che hanno permesso la realizzazione dei risultati che qui presento.

Sono inoltre riconoscente a Clément Aubert, dottorando presso LIPN, per le utili discussioni sull'ottimalità e le famiglie di Lévy, e a Thomas Levantis, studente *ENS Lyon* e stagista presso il LIPN, per la breve, ma intensa, collaborazione alle fasi centrali della dimostrazione del risultato di complessità. Un grazie va anche ad Ugo Dal Lago, che mi ha illuminato su alcuni dettagli del suo recente citato lavoro, e ad Andrea Asperti, con cui ho avuto il piacere di una breve discussione sull'ottimalità e i suoi possibili sviluppi.

A coloro i quali, nel lavoro che qui presento, mi hanno sostenuto e accompagnato in modi diversi, ma sempre preziosi, devo ringraziamenti personali, che avrò cura di fare personalmente.

Bibliografia

Asperti A. (1996). On the complexity of beta-reduction. In *Proceedings of the 23rd ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pp. 110–118. ACM. 4.3.1

Asperti A. (1998). Light affine logic. In *Logic in Computer Science, 13th International Symposium, Proceedings*, pp. 300–308. IEEE Computer Society. 4.2, 5.1

Asperti A.; Guerrini S. (1998). *The Optimal Implementation of Functional Programming Languages*. Cambridge University Press. 3, 3.3, 4.1, 6.1.2

Asperti A.; Mairson H. G. (1998). Parallel beta reduction is not elementary recursive. In *Conference record of POPL '98*, pp. 303–315. ACM Press. 4.3.1

Asperti A.; Roversi L. (2002). Intuitionistic light affine logic. *ACM Transactions on Computational Logic (TOCL)*, **3** (1), 137–175. 5.4

Asperti A.; Coppola P.; Martini S. (2004). (Optimal) Duplication is not elementary recursive. *Information and Computation*, **193**. 4.3.2

Baillot P.; Coppola P.; Dal Lago U. (2007). Light logics and optimal reduction: Completeness and complexity. In *Logic in Computer Science, 22nd International Symposium, Proceedings*, pp. 421–430. IEEE Computer Society. 7.4.1

Baillot P.; Coppola P.; Dal Lago U. (2011). Light logics and optimal reduction: Completeness and complexity. *Information and Computation*, **209** (2), 118–142. 4.2, 4.3.3, 7.3, 8, 8, 9.2

Barendregt H. P. (1984). *The Lambda Calculus, its Syntax and Semantics, Revised second edition*. North-Holland. 2

Bellantoni S.; Cook S. (1992). A new recursion-theoretic characterization of the polytime functions. *Computational Complexity*, pp. 97–110. 5.1

Dal Lago U. (2009). Context semantics, linear logic, and computational complexity. *ACM Transactions on Computational Logic (TOCL)*, **10** (4), 1–32. 4.3.3

Danos V.; Joinet J.-B. (2003). Linear logic and elementary time. *Information and Computation*, **183**. 5.4

Danos V.; Regnier L. (1989). The structure of multiplicatives. *Archive for Mathematical Logic*, **28** (3), 181–203. 10.1007/BF01622878. 5.7

Girard J.-Y. (1987). Linear logic. *Theoretical computer science*, **50** (1), 1–101. 5.1

Girard J.-Y. (1995). Light linear logic. In *Logic and computational complexity*, pp. 145–176. Springer. 5.1, 5.4

Gonthier G.; Abadi M.; Lévy J.-J. (1992). The geometry of optimal lambda reduction. In *Proceedings of the 19th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, POPL '92, pp. 15–26. ACM. 3, 4.1, 4.2

Guerrini S. (1999). A general theory of sharing graphs. *TCS: Theoretical Computer Science*, **227**. 4.1, 6.1.2

Guerrini S.; Martini S.; Masini A. (2000). Proof nets, garbage, and computations. *Theoretical computer science*, **253** (2), 185–237. 4.1, 4.2, 4.3.3

Guerrini S.; Martini S.; Masini A. (2003). Coherence for sharing proof-nets. *Theoretical computer science*, **294** (3), 379–409. 4.1, 4.2, 4.3.3

Hofmann M. (2000). Programming languages capturing complexity classes. *SIGACT News*, **31** (1), 31–42. 1

Lamping J. (1989). An algorithm for optimal lambda calculus reduction. In *17th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pp. 16–30. ACM. 3, 4.2

Lawall J. L.; Mairson H. G. (1996). Optimality and inefficiency : What isn't a cost model of the lambda calculus? In *ACM International Conference on Functional Programming*, pp. 92–101. 4.3.1

Leivant D. (1994). Ramified recurrence and computational complexity i: Word recurrence and poly-time. *Feasible Mathematics II*, pp. 320–343. 5.1

Lévy J.-J. (1978). *Réductions Correctes et Optimales dans le Lambda Calcul*. Tesi di Dottorato di Ricerca, Université Paris VII. 2, 4.3.1

Statman R. (1979). The typed lambda-calculus is not elementary recursive. *Theoretical Computer Science*, **9**, 73–81. 4.3.1

Terui K. (2007). Light affine lambda calculus and polynomial time strong normalization. *Archive for Mathematical Logic*, **46** (3-4), 253–280. 5.4, 9.1

Wadsworth C. (1971). *Semantics and pragmatics of the lambda-calculus*. Tesi di Dottorato di Ricerca, University of Oxford. 3