UNIVERSITÀ DEGLI STUDI ROMA TRE
FACOLTÀ DI SCIENZE M.F.N.

Sintesi della Tesi di Laurea in Matematica
presentata da

Francesca Cavaliere

Il teorema di Böhm nel λ-calcolo

Relatore

Prof. Lorenzo Tortora de Falco

Il Candidato | Il Relatore

ANNO ACCADEMICO 2003 - 2004
Febbraio 2005

Classificazione AMS :
Parole Chiave:

Il λ-calcolo è un sistema formale che pone la propria attenzione verso le funzioni, i metodi per calcolare e le regole di computazione, in definitiva verso il concetto di algoritmo che è poi centrale nello studio dell'informatica.
È stato introdotto negli anni '30 da Alonzo Church e Stephen Cole Kleene con lo scopo di costruire un sistema formale completo che servisse da base ai matematici.
Il lavoro di Moses Schönfinkel, nell'ambito delle discussioni sui fondamenti della matematica che caratterizzano gli inizi del XX secolo e più precisamente del programma hilbertiano mette in evidenza il ruolo centrale del concetto di funzione e delle regole di computazione che servono a definire le funzioni. Schönfinkel intendeva definire una collezione finita di termini (i combinatori) e di regole di computazione che per mezzo dell'applicazione funzionale consentissero di ottenere qualunque altro termine funzionale facendo uso di un linguaggio estremamente semplice in cui non ci fosse la necessità di variabili. A partire dal 1932, Church sviluppò l'idea di Schönfinkel introducendo le variabili e un operatore di astrazione che consente di comprenderne meglio il ruolo. Malgrado il lavoro non raggiunse gli scopi per i quali era stato iniziato, la teoria delle funzioni sviluppata in sé possiede senso e coerenza, e nel 1941 Church pubblica *The Calculi of λ-conversion* ([Chu41]). Nel 1943 assieme a Stephen C. Kleene sviluppa una nozione di computabilità (la Church-Kleene-computabilità) basata sul λ-calcolo.

Il λ-calcolo ha profondamente influenzato i linguaggi di programmazione funzionale. Può essere considerato il più semplice linguaggio universale di programmazione nel senso che ogni funzione calcolabile può essere espressa e calcolata tramite semplici sostituzioni di sequenze di simboli. Le espressioni del λ-calcolo sono sia funzioni che argomenti. Ogni espressione rappresenta una funzione ad un argomento e l'argomento di tale funzione, a sua volta, può essere una funzione ad un argomento il cui valore è ancora una funzione.

Ad esempio la funzione $f(x) = x + 2$ può essere espressa nel λ-calcolo come $\lambda\, x.x + 2$ dove $\lambda\, x$ dichiara la variabile di $f(x)$. Il valore $f(3)$, ad esempio, può essere scritto come l'applicazione di 3 a $\lambda\, x.x + 2$ ovvero $((\lambda\, x.x + 2)3)$, otteniamo infatti sostituendo 3 a x, il valore $3 + 2$.
La funzione $\lambda\, x.x + 2$ può essere, a sua volta, argomento di un λ-termine.
Ad esempio $\lambda\, x.x + 2$ applicata a $\lambda\, f\lambda\, x.f(fx)$ itera due volte $\lambda\, x.x + 2$

$$(\lambda\, f\lambda\, x.f(fx))(\lambda\, x.x + 2) \rightarrow \lambda\, x.(x + 2) + 2$$

ottenendo quindi la funzione $f(x) = x + 4$.

Nel λ-calcolo una funzione di due variabili è rappresentata da una funzione ad un argomento che restituisce una funzione ad un argomento, ovvero il λ-calcolo è sequenziale: una funzione a n variabili è espressa da una sequenza di n funzioni ad una variabile. Ad esempio, la funzione $f(x, y) = x + y$ può essere scritta come $\lambda\, xy.x + y$.

In questa tesi siamo partiti dall'analisi del λ-calcolo e siamo giunti ad un importante teorema: il teorema di Böhm. Il nostro obiettivo principale è stato quello di dimostrare tale teorema e di conoscere le conseguenze della sua applicazione all'insieme dei λ-termini.

Nel primo capitolo abbiamo descritto il modello del λ-calcolo definendo i λ-termini e le operazioni che possiamo effettuare su essi: la sostituzione, la α-equivalenza, la β-riduzione e la η-riduzione.
L'insieme dei λ-termini viene indicato con Λ e si ha $\Lambda \subset A^*$ dove A^* è l'insieme di tutte le combinazioni finite di elementi dell'alfabeto $A = \{\, (\, ,\,)\, ,\, \lambda\, \} \cup V$ in cui V è un insieme numerabile ed è costituito da variabili $x, y, z \dots$
Le regole da applicare per ottenere Λ sono le seguenti (*def.* 1.1.1, *pag.* 7):

- *una variabile* $x \in \Lambda$;

- *se* $M, N \in \Lambda$ *allora* $(MN) \in \Lambda$ *(applicazione);*

- *se* $M \in \Lambda$ *e* $x \in V$ *allora* $(\lambda\, x.M) \in \Lambda$ *(astrazione)*

Λ può essere descritto in modo più breve dalla seguente notazione:

$$T := x\,|(\lambda\, x.T)|\,(TT).$$

La relazione $\equiv$ indica l'uguaglianza sintattica tra due termini.
L'operazione di base nel λ-calcolo è la *sostituzione*.
Se $M, N_1, \ldots, N_n$ sono dei λ-termini e $x_1, \ldots, x_n$ variabili distinte, il termine $M[\vec{x} := \vec{N}]$ è il risultato che si ottiene sostituendo N_i ad ogni occorrenza libera di x_i in M per $1 \leqslant i \leqslant n$.
Tale termine è definito per induzione sulla lunghezza di M (*def.* 1.1.3, *pag.* 9):

1. *se* $M \in V$ *abbiamo i seguenti casi:*

   (a) *se* $M \equiv x_i$ $(1 \leqslant i \leqslant n)$, $M[\vec{x} := \vec{N}] \equiv N_i$;

   (b) *se* M *è una variabile diversa da* x_i, $M[\vec{x} := \vec{N}] \equiv M$;

2. *se* $M \equiv PQ$ *con* $P, Q \in \Lambda$,

$$M[\vec{x} := \vec{N}] \equiv P[\vec{x} := \vec{N}]Q[\vec{x} := \vec{N}];$$

3. *se* $M \equiv \lambda\, y.P$ *abbiamo i seguenti casi:*

   (a) *se* $y \neq x_i$ $(1 \leqslant i \leqslant n)$, $M[\vec{x} := \vec{N}] \equiv \lambda\, y.P[\vec{x} := \vec{N}]$;

   (b) *se* $y = x_i$ *per qualche i, allora*
   $M[\vec{x} := \vec{N}] \equiv$
   $\lambda\, y.P[x_1 := N_1, \ldots x_{i-1} := N_{i-1}, x_{i+1} := N_{i+1}, \ldots, x_n := N_n]$.

La sostituzione deve tenere conto delle variabili libere e vincolate, in particolare le variabili libere di un argomento non possono diventare vincolate tramite la sostituzione. Per poter distinguere le variabili vincolate della funzione da quelle dell'argomento si introduce la α-equivalenza (*def.* 1.1.5, *pag.* 11):

- *se M è una variabile, $M \simeq_\alpha N$ se e solo se $M \equiv N$;*
- *se $M \equiv PQ$, $M \simeq_\alpha N$ se e solo se $N \equiv P'Q'$, con $P \simeq_\alpha P'$ e $Q \simeq_\alpha Q'$.*
- *se $M \equiv \lambda\, x.P$, $M \simeq_\alpha N$ se e solo se $N \equiv \lambda\, x'.P'$,*
  con $P[x := y] \simeq_\alpha P'[x' := y]$, per ogni variabile y salvo un numero finito.

Infine la computazione dell'applicazione di una funzione $\lambda\, x.M$ ad un argomento N è formalizzata dalla definizione di β-riduzione (*def.* 1.1.6, *pag.* 12):

(i) *se M è una variabile, allora $M\, \beta_0\, M'$ è falsa per ogni M';*

(ii) *se $M \equiv \lambda\, x.N$, allora $M\, \beta_0\, M'$ sse $M' \equiv \lambda\, x.N'$ con $N\, \beta_0\, N'$;*

(iii) *se $M \equiv NL$, allora $M\, \beta_0\, M'$ sse:*
$M' \equiv NL'$ con $L\, \beta_0\, L'$, oppure
$M' \equiv N'L$ con $N\, \beta_0\, N'$, oppure
$N \equiv \lambda\, x.P$ e $M' \equiv P[x := L]$, rinominando opportunamente le variabili in modo che le occorrenze libere di L non vengano legate tramite tale sostituzione.

La β-equivalenza è la più piccola relazione binaria β su Λ, riflessiva, transitiva e che contiene β_0.
Abbiamo inoltre introdotto la η-equivalenza.

La η-riduzione rappresenta il principio di estensionalità: due termini che rappresentano la stessa funzione sono identici. Se $y \notin FV(M)$, abbiamo che $(\lambda\, y.My)x$ si β-riduce a Mx, quindi $(\lambda\, y.My)$ e M hanno lo stesso valore su ogni punto x, cioè rappresentano la stessa funzione. La definizione formale è (*def.* 1.1.8, *pag.* 15):

(i) *se* M *è una variabile, allora* $M\,\eta_0\,M'$ *è falsa per ogni* M';

(ii) *se* $M \equiv \lambda\, x.N$, *allora* $M\,\eta_0\,M'$ *se e solo se*

$M' \equiv \lambda\, x.N'$ *con* $N\,\eta_0\,N'$ *oppure*

$N \equiv M'x$, *con* x *non libera in* M';

(iii) *se* $M \equiv NL$, *allora* $M\,\eta_0\,M'$ *se e solo se:*

$M' \equiv NL'$ *con* $L\,\eta_0\,L'$ *oppure*

$M' \equiv N'L$ *con* $N\,\eta_0\,N'$

La η-equivalenza è la più piccola relazione binaria η su Λ, riflessiva, transitiva e che contiene η_0. Un termine della forma $(\lambda\, x.M)N$ è detto redex e $M[x := N]$ contratto; un termine della forma $\lambda\, x.Mx$ è detto η-redex quando x non compare libera in M.
Grazie alla β-riduzione abbiamo potuto introdurre la nozione di termine normalizzabile.
Se consideriamo un termine che contiene più redex, possiamo ridurlo a termini differenti scegliendo di ridurre un redex piuttosto che un altro. Un importante problema è la confluenza di queste riduzioni e cioè, se pur utilizzando riduzioni diverse, si possa sempre arrivare ad uno stesso risultato. Il teorema di Church-Rosser dimostra la confluenza della β-riduzione (*proprietà di Church-Rosser*), ovvero se $M \rightarrow_\beta M'$ e $N \rightarrow_\beta N'$ allora esiste un λ-termine L tale che $M' \rightarrow_\beta L$ e $N' \rightarrow_\beta L$.

Teorema 1.2.1 (Church-Rosser). *La β-conversione ha la proprietà di Church-Rosser.*

Si può dimostrare che anche la η-conversione e la $\beta\eta$-conversione hanno tale proprietà.
Vi sono termini che possono essere ridotti in forma normale e altri per cui ciò non è possibile. Abbiamo concluso il capitolo con il teorema di normalizzazione, dimostrando che esiste nel λ-calcolo una strategia di riduzione "sicura" ovvero che se esiste una forma normale per un dato termine M, allora tale procedura permette di raggiungerla. Tale strategia viene chiamata *riduzione sinistra* e consiste nel ridurre sempre il redex più a sinistra nel termine.

Teorema 1.3.3 (Normalizzazione). *Un λ-termine M è normalizzabile se e solo se la riduzione sinistra di M termina.*

Nel secondo capitolo abbiamo descritto altri due modelli di calcolo: le macchine di Turing e l'insieme delle funzioni ricorsive. La prima parte del capitolo è dedicata alla descrizione della macchina di Turing. Abbiamo dato la nozione di calcolabilità secondo Turing, e quindi di funzione Turing-calcolabile, e alcuni esempi per comprenderne meglio il funzionamento.
La seconda parte del capitolo riguarda le funzioni ricorsive. Partendo dalle funzioni di base si ottiene, tramite le operazioni di composizione e ricorsione, l'insieme delle funzioni ricorsive primitive. Aggiungendo a tale insieme un operatore μ, lo schema di minimalizzazione, si ottengono gli insiemi delle funzioni ricorsive parziali e delle funzioni ricorsive totali. Abbiamo inoltre considerato la funzione di Ackermann e dimostrato che è una funzione effettivamente calcolabile ma non ricorsiva primitiva.
L'ultima parte del capitolo è dedicata alla λ-definibilità. Tramite gli interi di Church, (ovvero i λ-termini che rappresentano gli interi e vengono indicati

con $\underline{1}, \underline{2}, \underline{3} \ldots$) e i booleani (**Vero** $\equiv \lambda\, xy.x$ e **Falso** $\equiv \lambda\, xy.y$) abbiamo dato la nozione di funzione λ-definibile (*def.* 2.3.1, *pag.* 42).

Una funzione parziale $\varphi : \mathbb{N}^p \to \mathbb{N}$ *è* λ-definibile *se esiste un termine* M_φ *tale che per ogni* $(n_1, \ldots, n_p) \in \mathbb{N}^p$ *si ha:*
se $\varphi(n_1, \ldots, n_p)$ *è definita, allora* $M_\varphi \underline{n_1} \ldots \underline{n_p} \to_\beta \underline{\varphi(n_1, \ldots, n_p)}$;
se $\varphi(n_1, \ldots, n_p)$ *non è definita, allora* $M_\varphi \underline{n_1} \ldots \underline{n_p}$ *non è risolubile.*
Diremo che M_φ *rappresenta la funzione* φ *e che* φ *è rappresentabile dal termine* M_φ.

In [Kle36] Kleene dimostra il seguente risultato:

Teorema 2.3.1 (Kleene). *Una funzione parziale è ricorsiva se e solo se è λ-definibile.*

Noi abbiamo dimostrato il lato "ricorsiva parziale $\Rightarrow$ λ-definibile" del teorema. Per tale dimostrazione abbiamo introdotto dei particolari λ-termini: i combinatori di punto fisso, che hanno un ruolo simile all'operatore μ nella teoria della ricorsività.
Il capitolo si conclude con la *Tesi di Church*, con la quale si afferma che i modelli di calcolo descritti, tutti equivalenti, rappresentano la classe delle funzioni calcolabili:

Tesi di Church: *La classe delle funzioni "intuitivamente calcolabili" coincide con la classe delle funzioni Turing calcolabili.*

Il capitolo 3 è stato interamente dedicato al teorema di Böhm.
Tale teorema afferma che se M e N sono due λ-termini in forma $\beta\eta$-normale distinta allora è sempre possibile separarli, cioè renderli equivalenti a due variabili distinte. Per separare M ed N si costruisce un contesto $C[\ ]$ che è un λ-termine con dei buchi all'interno.
Abbiamo innanzitutto mostrato che esistono tre versioni del teorema: una

debole e due forti, quest'ultime sono equivalenti tra loro e implicano la versione debole. Se M è un termine, indichiamo con $M\downarrow$ che M è normalizzabile e con $M\uparrow$ che M non è normalizzabile cioè non possiede una forma normale. Le versioni forti sono le seguenti:

- Siano M, N due termini chiusi in forma normale distinta, allora esistono $L_1, \ldots, L_n$ termini ($n \geqslant 0$), tali che

$$ML_1 \ldots L_n \to_\beta \mathbf{V},$$

$$NL_1 \ldots L_n \to_\beta \mathbf{F}$$

  dove $\mathbf{V} \equiv \lambda\, xy.x$ e $\mathbf{F} \equiv \lambda\, xy.y$.

- Siano M, N due termini chiusi in forma normale distinta, P e Q due termini arbitrari, allora esistono $L_1, \ldots, L_n$ termini ($n \geqslant 0$), tali che

$$ML_1 \ldots L_n \to_\beta P,$$

$$NL_1 \ldots L_n \to_\beta Q.$$

La versione debole é:

- Siano M, N due termini chiusi in forma normale distinta, allora esistono $L_1, \ldots, L_n$ termini ($n \geqslant 0$), tali che

$$ML_1 \ldots L_n \downarrow$$

$$NL_1 \ldots L_n \uparrow$$

L'enunciato del teorema che abbiamo dimostrato è:

Teorema 3.6.1 (Böhm). *Siano M, N due forme $\beta\eta$-normali distinte e x, y due variabili distinte. Allora esiste un contesto $C[\ \ ]$ tale che $C[M] \equiv x$ e $C[N] \equiv y$.*

Il teorema di Böhm viene dimostrato per induzione sulla profondità delle forme normali. La dimostrazione risulta particolarmente chiara se si rappresenta una forma normale come un albero. Una forma normale è infatti ”un inscatolamento” di forme normali.
Tale albero è stato definito in [Bar84] come *albero di Böhm* e se M è un termine viene indicato con $BT(M)$. È così definito (*def.* 3.3.1, *pag.* 56):

Sia $\Sigma = \{\lambda\, x_1 \ldots x_n.y \mid n \in \mathbb{N}, x_1, \ldots, x_n, y$ *variabili*$\}$, *allora* BT(M) *è un albero etichettato da* Σ *definito come segue:*

$BT(M) = \lambda\, x_1 \ldots x_n.y$	*se* M *ha forma normale di testa* $\lambda\, x_1 \ldots x_n.y$
$BT(M) = \lambda\, x_1 \ldots x_n.y$	*se* M *ha forma normale di testa* $\lambda\, x_1 \ldots x_n.y M_1 \ldots M_m$
$BT(M_1) \;\ldots\; BT(M_m)$	

Esempio. L'albero di $\lambda\, x_1x_2x_3.x_1(x_2(\lambda\, y.y))(x_2x_3)x_3$ è:

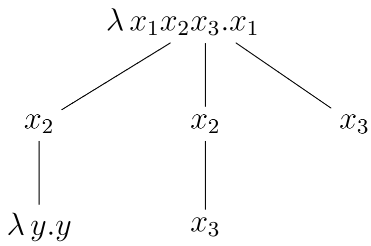

Figura 1: Albero di Böhm di $M \equiv \lambda\, x_1x_2x_3.x_1(x_2(\lambda\, y.y))(x_2x_3)x_3$

Per la dimostrazione del teorema è fondamentale introdurre delle operazioni sopra i λ-termini: *le trasformazioni di Böhm.*
Poiché il teorema considera due termini in forma normale distinta, ad esem-

pio $M \equiv \lambda x_1 \dots x_n.yM_1 \dots M_m$ e $N \equiv \lambda x_1 \dots x_{n'}.y'N_1 \dots N_{m'}$, è necessario studiare la struttura sintattica dei due termini. Se $M \not\simeq_\beta \eta N$, la differenza potrebbe essere in testa, ovvero sulla radice dell'albero, oppure in uno dei loro argomenti. Per il primo caso abbiamo dato la seguente definizione:

Se M ed N sono due termini in forma normale di testa, cioè

$$M \equiv \lambda x_1 \dots x_n.yM_1 \dots M_m \quad e \quad N \equiv \lambda x_1 \dots x_{n'}.y'N_1 \dots N_{m'}$$

diciamo che M è $\sim$-equivalente *a N e scriviamo*

$$M \sim N \quad sse \quad y \equiv y' \quad e \quad n - m = n' - m'$$

e il seguente risultato:

Lemma 3.4.1. *Siano M, N due λ-termini normalizzabili.*
Se $M \nsim N$ allora esiste una trasformazione π tale che

$$M^\pi \simeq_\beta \boldsymbol{V} \quad e \quad N^\pi \simeq_\beta \boldsymbol{F}.$$

Nel caso in cui la differenza sia in uno degli argomenti di M e N abbiamo utilizzato la tecnica del Böhm-out che consiste nel costruire una sequenza di termini che, se passati come argomenti a due forme normali distinte, estraggono un sottotermine per il quale essi differiscono. Questo è ottenuto legando, in successione, le variabili di testa con degli appropriati *selezionatori* così da poter percorrere il cammino dalla radice fino al nodo desiderato degli alberi di Böhm associati ai due λ-termini.
Abbiamo ottenuto il risultato seguente:

Lemma 3.5.1. *Siano M ed N due forme normali di testa distinte e tali che $M \sim N$. Siano M_i e N_i rispettivamente l'i-esimo argomento di M ed N.*
Esiste una trasformazione π tale che, per ogni α, se

$$M_{i*\alpha} \nsim N_{i*\alpha} \quad allora \quad (M^\pi)_\alpha \nsim (N^\pi)_\alpha.$$

In entrambi i casi abbiamo visto che è possibile distinguere i due termini: nel primo caso siamo riusciti a separare i due termini rendendoli equivalenti rispettivamente a **V** e a **F**, nel secondo siamo riusciti a portare la $\beta\eta$-differenza in testa, e quindi a ricondurci al primo caso.

Nel capitolo 4 concludiamo studiando separazioni analoghe a quella introdotta dal teorema di Böhm, per λ-termini anche non normalizzabili.
Abbiamo considerato l'equivalenza osservazionale ovvero un'equivalenza tra i λ-termini che hanno comportamento analogo quando vengono inseriti nello stesso contesto. In particolare abbiamo considerato tre diverse equivalenze osservazionali:

1) osservando se l'applicazione di due λ-termini dentro un contesto ha o meno forma normale si definisce l'equivalenza normale;

2) osservando se l'applicazione di due λ-termini dentro un contesto ha o meno forma normale di testa si definisce l'equivalenza di testa;

3) osservando se l'applicazione di due λ-termini dentro un contesto ha o meno forma normale di testa debole si definisce l'equivalenza di testa debole.

Abbiamo esteso la definizione di alberi di Böhm anche a termini non normalizzabili ammettendo rami infiniti o nodi con etichetta indefinita (*def.* 4.2.3 *pag.* 76 e *def.* 4.3.2 *pag.* 78). In seguito abbiamo enunciato il teorema di Hyland per affermare che l'equivalenza osservazionale normale coincide con l'equivalenza sintattica degli η-alberi di Böhm e il teorema di Wadsworth per affermare che l'equivalenza osservazionale di testa coincide con l'equivalenza sintattica degli η_∞-alberi di Böhm. Abbiamo concluso il capitolo dimostrando che l'equivalenza osservazionale di testa debole è quella che distingue più delle altre.

Bibliografia

[Bar84] H. P. Barendregt. *The lambda calculus. Its syntax and semantics*, volume 103 of Studies in Logic and the Foundations of Mathematics. North-Holland Publishing Co., Amsterdam, revised edition, 1984.

[BCD83] H. P. Barendregt, M. Coppo, and M. Dezani-Ciancaglini. *A filter lambda model and the completeness of type assignment*. J. Symbolic Logic, 48(4):931–940 (1984), 1983.

[Böh68] C. Böhm. *Alcune proprietà delle forme β-η-normali nel λ-K-calcolo*. Pubblicazioni dell'IAC, 696:1–19, 1968.

[Chu32] Alonzo Church. A set of postulates for the foundation of logic. *Ann. of Math. (2)*, 33(2):346–366, 1932.

[Chu33] Alonzo Church. A set of postulates for the foundation of logic. *Ann. of Math. (2)*, 34(4):839–864, 1933.

[Chu41] A. Church. *The Calculi of Lambda-Conversion*. Annals of Mathematics Studies, no. 6. Princeton University Press, Princeton, N. J., 1941.

[DG01] M. Dezani-Ciancaglini and E. Giovannetti. *From Böhm's Theorem to Observational Equivalences: an Informal Account*. Electr. Notes Theor. Comput. Sci., 50(2), 2001.

[DIV98] M. Dezani-Ciancaglini, B. Intrigila, and M. Venturini-Zilli. *Böhm's theorem for Böhm trees.* In Theoretical computer science (Prato, 1998), pages 1–23. World Sci. Publishing, River Edge, NJ, 1998.

[HS86] J. R. Hindley and J. P. Seldin. *Introduction to combinators and λ-calculus*, volume 1 of London Mathematical Society Student Texts. Cambridge University Press, Cambridge, 1986.

[Hue93] G. Huet. *An analysis of Böhm's theorem.* Theoret. Comput. Sci.,121:145–167, 1993.

[Hyl76] M. Hyland. *A syntactic characterization of the equality in some models for the lambda-calculus.* J. London Math. Soc. (2), 12(3):361–370, 1975/76.

[Kle36] S. C. Kleene. *λ-definability and recursiveness.* Duke Math. J., 2:340–353, 1936.

[Kri90] J.-L. Krivine. *Lambda-calcul. Types et modèles.* Études et Recherches en Informatique. Masson, Paris, 1990.

[Mit79] G. Mitschke. *The standardization theorem for λ calculus.* Z. Math. Logik Grundlag. Math., 25(1):29–31, 1979.

[Mor68] J. H. Morris. *λ-calculus models of programming languages.* PhD thesis, M.I.T., 1968.

[Sch24] M. Schönfinkel. *Über die Bausteine der mathematischen Logik.* Math. Annalen (92):305–316, 1924.

[Tur36] A. M. Turing. *On computable numbers with an application to the Entscheidungsproblem.* Proc. London Math. Soc. (42):230–265, 1936.

[Tur37] A. M. Turing. *Computability and λ-definability.* J. Symbolic Logic (2):153–163, 1937.

[Wad76] C. P. Wadsworth. *The relation between computational and denotational properties for Scott's D_∞-models of the lambda-calculus.* SIAM J. Comput., 5(3):488–521, 1976.