Chapter 6

Linear λ-Calculus

In philosophy we distinguish between the notion of *analytic* and *synthetic* judgment [ML94], a terminology which goes back to Kant. Briefly, an analytic judgment can be seen to be evident by virtue of the terms contained in it. A synthetic judgment, on the other hand, requires to go beyond the judgment itself to find evidence for it. The various judgments of truth we have considered such as $\Gamma; \Delta \vdash A$ *true* are *synthetic* because we need the derivation as evidence external to the judgment. We can contrast this with the judgment A *prop* which is *analytic*: an analysis of A itself is sufficient to see if it is a proposition.

It is important to recognize analytic judgments because we do not need to communicate external evidence for them if we want to convince someone of it. The judgment itself carries the evidence. A standard way to convert a synthetic to a corresponding analytic judgment is to enrich it with a term that carries enough information to reconstruct its deduction. We refer to such objects as *proof terms* when they are used to establish the truth of a proposition. There still is a fair amount of latitude in designing proof terms, but with a few additional requirements discussed below they are essentially determined by the structure of the inference rules.

From our intuitionistic point of view it should not be surprising that such proof terms describe constructions. For example, a proof of $A \multimap B$ describes a construction for achieving the goal B given resource A. This can be seen as a plan or a program. In (unrestricted) intuitionistic logic, the corresponding observation that proofs are related to *functional* programs via the *Curry-Howard isomorphism* has been made by [CF58] and [How80]. Howard observed that there is a bijective correspondence between proofs in intuitionistic propositional natural deduction and simply-typed λ-terms. A related observation on proof in combinatory logic had been made previously by Curry [CF58].

A generalization of this observation to include quantifiers gives rise to the rich field of type theory, which we will analyze in Chapter **??**. Here we study the basic correspondence, extended to the case of linear logic.

A linear λ-calculus of proof terms will be useful for us in various circumstances. First of all, it gives a compact and faithful representation of proofs as terms. Proof checking is reduced to type-checking in a λ-calculus. For example, if we do not trust the implementation of our theorem prover, we can instrument it to generate proof terms which can be verified independently. In this scenario we are just exploiting that validity of proof terms is an analytic judgment. Secondly, the terms in the λ-calculus provide the core of a functional language with an expressive type system, in which statements such as "*this function will use its argument exactly once*" can be formally expressed and checked. It turns out that such properties can exploited for introducing imperative (state-related) concepts into functional programming [Hof00b], structural complexity theory [Hof00a, AS01], or analysis of memory allocation [WW01] Thirdly, linear λ-terms can serve as an expressive representation language within a *logical framework*, a general meta-language for the formalization of deductive systems.

6.1 Proof Terms

We now assign proof terms to the system of linear natural deduction. Our main criterion for the design of the proof term language is that the proof terms should reflect the structure of the deduction as closely as possible. Moreover, we would like every valid proof term to uniquely determine a natural deduction. Because of weakening for unrestricted hypotheses and the presence of $\top$, this strong property will fail, but a slightly weaker and, from the practical point of view, sufficient property holds. Under the Curry-Howard isomorphism, a proposition corresponds to a type in the proof term calculus. We will there call a proof term *well-typed* if it represents a deduction.

The proof term assignment is defined via the judgment $\Gamma; \Delta \vdash M : A$, where each formula in Γ and Δ is labelled. We also use $M \longrightarrow_\beta M'$ for the local reduction and $M : A \longrightarrow_\eta M'$ for the local expansion, both expressed on proof terms. The type on the left-hand side of the expansion reminds is a reminder that this rule only applies to term of the given type (contexts are elided here).

Hypotheses. We use the label of the hypotheses as the name for a variable in the proof terms. There are no reductions or expansions specific to variables, although variables of non-atomic type may be expanded by the later rules.

$$\frac{}{\Gamma; u{:}A \vdash u : A}\,u \qquad \frac{}{(\Gamma, v{:}A); \cdot \vdash v : A}\,u$$

Recall that we take exchange for granted so that in the rule for unrestricted hypotheses, $v{:}A$ could occur anywhere.

Multiplicative Connectives. Linear implication corresponds to a *linear function types* with corresponding linear abstraction and application. We distinguish them from unrestricted abstraction and application by a "hat". In certain circumstances, this may be unnecessary, but here we want to reflect the proof structure as directly as possible.

$$\frac{\Gamma; (\Delta, u{:}A) \vdash M : B}{\Gamma; \Delta \vdash \hat{\lambda} u{:}A.\ M : A \multimap B} \multimap I$$

$$\frac{\Gamma; \Delta \vdash M : A \multimap B \qquad \Gamma; \Delta' \vdash N : A}{\Gamma; (\Delta, \Delta') \vdash M \,\hat{}\, N : B} \multimap E$$

$$\begin{array}{rcl} (\hat{\lambda} w{:}A.\ M) \,\hat{}\, N & \longrightarrow_\beta & [N/w]M \\ M : A \multimap B & \longrightarrow_\eta & \hat{\lambda} w{:}A.\ M \,\hat{}\, w \end{array}$$

In the rules for the simultaneous conjunction, the proof term for the elimination inference is a let form which deconstructs a pair, naming the components. The linearity of the two new hypotheses means that the variables must both be used in M.

$$\frac{\Gamma; \Delta_1 \vdash M : A \qquad \Gamma; \Delta_2 \vdash N : B}{\Gamma; (\Delta_1, \Delta_2) \vdash M \otimes N : A \otimes B} \otimes I$$

$$\frac{\Gamma; \Delta \vdash M : A \otimes B \quad \Gamma; (\Delta', u{:}A, w{:}B) \vdash N : C}{\Gamma; (\Delta, \Delta') \vdash \mathsf{let}\, u \otimes w = M \,\mathsf{in}\, N : C} \otimes E$$

The reduction and expansion mirror the local reduction and expansion for deduction as the level of proof terms. We do not reiterate them here, but simply give the proof term reduction.

$$\begin{array}{rcl} \mathsf{let}\, w_1 \otimes w_2 = M_1 \otimes M_2 \,\mathsf{in}\, N & \longrightarrow_\beta & [M_1/w_1, M_2/w_2]N \\ M : A \otimes B & \longrightarrow_\eta & \mathsf{let}\, w_1 \otimes w_2 = M \,\mathsf{in}\, w_1 \otimes w_2 \end{array}$$

The unit type allows us to consume linear hypotheses without introducing new linear ones.

$$\frac{}{\Gamma; \cdot \vdash \star : \mathbf{1}} \mathbf{1}I \qquad \frac{\Gamma; \Delta \vdash M : \mathbf{1} \qquad \Gamma; \Delta' \vdash N : C}{\Gamma; (\Delta, \Delta') \vdash \mathsf{let}\, \star = M \,\mathsf{in}\, N : C} \mathbf{1}E$$

$$\begin{array}{rcl} \mathsf{let}\, \star = M \,\mathsf{in}\, N & \longrightarrow_\beta & N \\ M : \star & \longrightarrow_\eta & \mathsf{let}\, \star = M \,\mathsf{in}\, \star \end{array}$$

Additive Connectives. As we have seen from the embedding of intuitionistic in linear logic, the simultaneous conjunction represents products from the simply-typed λ-calculus.

$$\frac{\Gamma; \Delta \vdash M : A \qquad \Gamma; \Delta \vdash N : B}{\Gamma; \Delta \vdash \langle M, N \rangle : A \& B} \& I$$

$$\frac{\Gamma; \Delta \vdash M : A \& B}{\Gamma; \Delta \vdash \mathsf{fst}\, M : A} \& E_L \qquad \frac{\Gamma; \Delta \vdash M : A \& B}{\Gamma; \Delta \vdash \mathsf{snd}\, M : B} \& E_R$$

The local reduction are also the familiar ones.

$$\begin{array}{rcl} \mathsf{fst}\,\langle M_1, M_2\rangle & \longrightarrow_\beta & M_1 \\ \mathsf{snd}\,\langle M_1, M_2\rangle & \longrightarrow_\beta & M_2 \\ M : A \& B & \longrightarrow_\eta & \langle \mathsf{fst}\,M, \mathsf{snd}\,M\rangle \end{array}$$

The additive unit corresponds to a unit type with no operations on it.

$$\frac{}{\Gamma;\Delta \vdash \langle\,\rangle : \top}\top\mathrm{I} \qquad \textit{No } \top \textit{ elimination}$$

The additive unit has no elimination and therefore no reduction. However, it still admits an expansion, which witnesses the local completeness of the rules.

$$M : \top \quad \longrightarrow_\eta \quad \langle\,\rangle$$

The disjunction (or *disjoint sum* when viewed as a type) uses injection and case as constructor and destructor forms, respectively. We annotated the injections with a type to preserve the property that any well-typed term has a unique type.

$$\frac{\Gamma;\Delta \vdash M : A}{\Gamma;\Delta \vdash \mathsf{inl}^B\,M : A \oplus B}\oplus\mathrm{I_L} \qquad \frac{\Gamma;\Delta \vdash M : B}{\Gamma;\Delta \vdash \mathsf{inr}^A\,M : A \oplus B}\oplus\mathrm{I_R}$$

$$\frac{\Gamma;\Delta \vdash M : A \oplus B \quad \Gamma;(\Delta', w_1{:}A) \vdash N_1 : C \quad \Gamma;(\Delta', w_2{:}B) \vdash N_2 : C}{\Gamma;(\Delta',\Delta) \vdash \mathsf{case}\,M\,\mathsf{of}\,\mathsf{inl}\,w_1 \Rightarrow N_1 \mid \mathsf{inr}\,w_2 \Rightarrow N_2 : C}\oplus\mathrm{E}^{w_1,w_2}$$

The reductions are just like the ones for disjoint sums in the simply-typed λ-caclulus.

$$\begin{array}{rcl} \mathsf{case}\,\mathsf{inl}^B\,M\,\mathsf{of}\,\mathsf{inl}\,w_1 \Rightarrow N_1 \mid \mathsf{inr}\,w_2 \Rightarrow N_2 & \longrightarrow_\beta & [M/w_1]N_1 \\ \mathsf{case}\,\mathsf{inr}^A\,M\,\mathsf{of}\,\mathsf{inl}\,w_1 \Rightarrow N_1 \mid \mathsf{inr}\,w_2 \Rightarrow N_2 & \longrightarrow_\beta & [M/w_2]N_2 \end{array}$$

$$M : A \oplus B \quad \longrightarrow_\eta \quad \mathsf{case}\,M\,\mathsf{of}\,\mathsf{inl}\,w_1 \Rightarrow \mathsf{inl}^B\,w_1 \mid \mathsf{inr}\,w_2 \Rightarrow \mathsf{inr}^A\,w_2$$

For the additive falsehood, there is no introduction rule. It corresponds to a *void type* without any values. Consequently, there is no reduction. Once again we annotate the abort constructor in order to guarantee uniqueness of types.

$$\textit{No } \mathbf{0} \textit{ introduction} \qquad \frac{\Gamma;\Delta \vdash M : \mathbf{0}}{\Gamma;(\Delta,\Delta') \vdash \mathsf{abort}^C\,M : C}\mathbf{0}\mathrm{E}$$

$$M : \mathbf{0} \quad \longrightarrow_\eta \quad \mathsf{abort}^{\mathbf{0}}\,M$$

Exponentials. Unrestricted implication corresponds to the usual *function type* from the simply-typed λ-calculus. For consistency, we will still write $A \supset B$ instead of $A \to B$, which is more common in λ-calculus. Note that the argument of an unrestricted application may not mention any linear variables.

$$\frac{(\Gamma, v{:}A); \Delta \vdash M : B}{\Gamma; \Delta \vdash \lambda v{:}A.\ M : A \supset B} \supset\text{I}$$

$$\frac{\Gamma; \Delta \vdash M : A \supset B \qquad \Gamma; \cdot \vdash N : A}{\Gamma; \Delta \vdash M\,N : B} \supset\text{E}$$

The reduction and expansion are the origin of the β and η rules names due to Church [Chu41].

$$\begin{array}{rcl} (\lambda v{:}A.\ M)\,N & \longrightarrow_\beta & [N/v]M \\ M : A \supset B & \longrightarrow_\eta & \lambda v{:}A.\ M\,v \end{array}$$

The rules for the *of course* operator allow us to name term of type $!A$ and use it freely in further computation.

$$\frac{\Gamma; \cdot \vdash M : A}{\Gamma; \cdot \vdash\, !M : !A} !\text{I} \qquad \frac{\Gamma; \Delta \vdash M : !A \qquad (\Gamma, v{:}A); \Delta' \vdash N : C}{\Gamma; (\Delta', \Delta) \vdash \mathsf{let}\ !v = M\ \mathsf{in}\ N : C} !\text{E}$$

$$\begin{array}{rcl} \mathsf{let}\ !v = !M\ \mathsf{in}\ N & \longrightarrow_\beta & [M/v]N \\ M : !A & \longrightarrow_\eta & \mathsf{let}\ !v = M\ \mathsf{in}\ !v \end{array}$$

Below is a summary of the linear λ-calculus with the β-reduction and η-expansion rules.

$$\begin{array}{lll} M ::= & u & \textit{Linear Variables} \\ & \mid \hat{\lambda} u{:}A.\ M \mid M_1 \hat{}\ M_2 & A \multimap B \\ & \mid M_1 \otimes M_2 \mid \mathsf{let}\ u_1 \otimes u_2 = M\ \mathsf{in}\ M' & A \otimes B \\ & \mid \star \mid \mathsf{let}\ \star = M\ \mathsf{in}\ M' & \mathbf{1} \\ & \mid \langle M_1, M_2 \rangle \mid \mathsf{fst}\ M_1 \mid \mathsf{snd}\ M_2 & A \& B \\ & \mid \langle\,\rangle & \top \\ & \mid \mathsf{inl}^B\ M \mid \mathsf{inr}^A\ M & A \oplus B \\ & \quad \mid (\mathsf{case} M\ \mathsf{of}\ \mathsf{inl}\ u_1 \Rightarrow M_1 \mid \mathsf{inr}\ u_2 \Rightarrow M_2) & \\ & \mid \mathsf{abort}^C\ M & \mathbf{0} \\ & \mid v & \textit{Unrestricted Variables} \\ & \mid \lambda v{:}A.\ M \mid M_1\, M_2 & A \supset B \\ & \mid\, !M \mid \mathsf{let}\ v = M\ \mathsf{in}\ M' & !A \end{array}$$

Below is a summary of the β-reduction rules, which correspond to local reductions of natural deductions.

$$
\begin{array}{rcll}
(\hat{\lambda} u{:}A.\ M)\,\hat{}\,N & \longrightarrow_\beta & [N/u]M & A \multimap B \\
\mathsf{let}\ u_1 \otimes u_2 = M_1 \otimes M_2\ \mathsf{in}\ N & \longrightarrow_\beta & [M_1/u_1, M_2/u_2]N & A \otimes B \\
\mathsf{let}\ \star = M\ \mathsf{in}\ N & \longrightarrow_\beta & N & \mathbf{1} \\
\mathsf{fst}\ \langle M_1, M_2 \rangle & \longrightarrow_\beta & M_1 & A \& B \\
\mathsf{snd}\ \langle M_1, M_2 \rangle & \longrightarrow_\beta & M_2 & \\
\textit{No } \top \textit{ reduction} & & & \\
\mathsf{case}\ \mathsf{inl}^B\ M\ \mathsf{of}\ \mathsf{inl}\ u_1 \Rightarrow N_1 \mid \mathsf{inr}\ u_2 \Rightarrow N_2 & \longrightarrow_\beta & [M/u_1]N_1 & A \oplus B \\
\mathsf{case}\ \mathsf{inr}^A\ M\ \mathsf{of}\ \mathsf{inl}\ u_1 \Rightarrow N_1 \mid \mathsf{inr}\ u_2 \Rightarrow N_2 & \longrightarrow_\beta & [M/u_1]N_2 & \\
\textit{No } \mathbf{0} \textit{ reduction} & & & \\
(\lambda v{:}A.\ M)\ N & \longrightarrow_\beta & [N/v]M & A \supset B \\
\mathsf{let}\ !v = !M\ \mathsf{in}\ N & \longrightarrow_\beta & [M/v]N & !A
\end{array}
$$

The substitution $[M/u]N$ and $[M/v]N$ assumes that there are no free variables in M which would be captured by a variables binding in N. We nonethless consider it a total function, since the capturing variable can always be renamed to avoid a conflict (see Exercise 6.3).

Next is a summary of the η-expansion rules, which correspond to local expansions of natural deductions.

$$
\begin{array}{rcl}
M : A \multimap B & \longrightarrow_\eta & \hat{\lambda} u{:}A.\ M\,\hat{}\,u \\
M : A \otimes B & \longrightarrow_\eta & \mathsf{let}\ u_1 \otimes u_2 = M\ \mathsf{in}\ u_1 \otimes u_2 \\
M : \star & \longrightarrow_\eta & \mathsf{let}\ \star = M\ \mathsf{in}\ \star \\
M : A \& B & \longrightarrow_\eta & \langle \mathsf{fst}\ M, \mathsf{snd}\ M \rangle \\
M : \top & \longrightarrow_\eta & \langle\,\rangle \\
M : A \oplus B & \longrightarrow_\eta & \mathsf{case}\ M\ \mathsf{of}\ \mathsf{inl}\ u_1 \Rightarrow \mathsf{inl}^B\ u_1 \mid \mathsf{inr}\ u_2 \Rightarrow \mathsf{inr}^A\ u_2 \\
M : \mathbf{0} & \longrightarrow_\eta & \mathsf{abort}^{\mathbf{0}}\ M \\
M : A \supset B & \longrightarrow_\eta & \lambda v{:}A.\ M\ v \\
M : !A & \longrightarrow_\eta & \mathsf{let}\ !v = M\ \mathsf{in}\ !v
\end{array}
$$

Note that there is an implicit assumption that the variables w and u in the cases for $A \multimap B$ and $A \supset B$ do not already occur in M: they are chosen to be new.

If $\mathcal{P}$ is a derivation of $\Gamma; \Delta \vdash M : A$ then we write erase($\mathcal{P}$) for the corresponding derivation of $\Gamma; \Delta \vdash A$ *true* where the proof term M has been erased from every judgment.

We have the following fundamental properties. Uniqueness, where claimed, holds only up to renaming of bound variables.

Theorem 6.1 (Properties of Proof Terms)

1. *If $\mathcal{P}$ is a derivation of $\Gamma; \Delta \vdash M : A$ then* erase($\mathcal{P}$) *is a derivation of $\Gamma; \Delta \vdash A$.*

2. *If $\mathcal{D}$ is a derivation of $\Gamma; \Delta \vdash A$ then there is a unique M and derivation $\mathcal{P}$ of $\Gamma; \Delta \vdash M : A$ such that* erase($\mathcal{P}$) $= \mathcal{D}$.

Proof: By straightforward inductions over the given derivations. □

Types are also unique for well-typed terms (see Exercise 6.1). Uniqueness of derivations fails, that is, a proof term does not uniquely determine its derivation, even under identical contexts. A simple counterexample is provided by the following two derivations (with the empty unrestricted context elided).

$$\dfrac{\dfrac{}{w{:}\top \vdash \langle\,\rangle : \top}\top I \quad \dfrac{}{\cdot \vdash \langle\,\rangle : \top}\top I}{w{:}\top \vdash \langle\,\rangle \otimes \langle\,\rangle : \top \otimes \top}\otimes I \qquad \dfrac{\dfrac{}{\cdot \vdash \langle\,\rangle : \top}\top I \quad \dfrac{}{w{:}\top \vdash \langle\,\rangle : \top}\top I}{w{:}\top \vdash \langle\,\rangle \otimes \langle\,\rangle : \top \otimes \top}\otimes I$$

It can be shown that linear hypotheses which are absorbed by $\top I$ are the only source of only ambiguity in the derivation. A similar ambiguity already exists in the sense that any proof term remains valid under weakening in the unrestricted context: whenever $\Gamma;\Delta \vdash M : A$ then $(\Gamma,\Gamma');\Delta \vdash M : A$. So this phenomenon is not new to the linear λ-calculus, and is in fact a useful identification of derivations which differ in "irrelevant" details, that is, unused or absorbed hypotheses.

The substitution principles on natural deductions can be expressed on proof terms. This is because the translations from natural deductions to proof terms and *vice versa* are *compositional*: uses of a hypothesis labelled u in natural deduction corresponds to an occurrence of a variable u in the proof term.

Lemma 6.2 (Substitution on Proof Terms)

1. *If* $\Gamma;(\Delta,w{:}A) \vdash N{:}C$ *and* $\Gamma;\Delta' \vdash M : A$, *then* $\Gamma;(\Delta,\Delta') \vdash [M/w]N : C$.
2. *If* $(\Gamma,u{:}A);\Delta \vdash N{:}C$ *and* $\Gamma;\cdot \vdash M : A$, *then* $\Gamma;\Delta \vdash [M/u]N : C$.

Proof: By induction on the structure of the first given derivation, using the property of exchange. □

We also have the property of weakening for unrestricted hypotheses. The substitution properties are the critical ingredient for the important *subject reduction* properties, which guarantee that the result of β-reducing a well-typed term will again be well-typed. The expansion rules also preserve types when invoked properly.

Theorem 6.3 (Subject Reduction and Expansion)

1. *If* $\Gamma;\Delta \vdash M : A$ *and* $M \longrightarrow_\beta M'$ *then* $\Gamma;\Delta \vdash M' : A$.
2. *If* $\Gamma;\Delta \vdash M : A$ *and* $M : A \longrightarrow_\eta M'$ *then* $\Gamma;\Delta \vdash M' : A$.

Proof: For subject reduction we examine each possible reduction rule, applying inversion to obtain the shape of the typing derivation. From this we either directly construct the typing derivation of M' or we appeal to the substitution lemma.

For subject expansion we directly construct the typing derivation for M' from the typing derivation of M. □

Note that the opposite of subject reduction does not hold: there are well-typed terms M' such that $M \longrightarrow_\beta M'$ and M is not well-typed (see Exercise 6.4).

Bibliography

[ABCJ94] D. Albrecht, F. Bäuerle, J. N. Crossley, and J. S. Jeavons. Curry-Howard terms for linear logic. In ??, editor, *Logic Colloquium '94*, pages ??–?? ??, 1994.

[Abr93] Samson Abramsky. Computational interpretations of linear logic. *Theoretical Computer Science*, 111:3–57, 1993.

[ACS98] Roberto Amadio, Ilaria Castellani, and Davide Sangiorgi. On bisimulations for the asynchrounous π-calculus. *Theoretical Computer Science*, 195(2):291–423, 1998.

[AK91] Hassan Aït-Kaci. *Warren's Abstract Machine: A Tutorial Reconstruction.* MIT Press, 1991.

[And92] Jean-Marc Andreoli. Logic programming with focusing proofs in linear logic. *Journal of Logic and Computation*, 2(3):197–347, 1992.

[AP91] J.-M. Andreoli and R. Pareschi. Logic programming with sequent systems: A linear logic approach. In P. Schröder-Heister, editor, *Proceedings of Workshop to Extensions of Logic Programming, Tübingen, 1989*, pages 1–30. Springer-Verlag LNAI 475, 1991.

[AS01] Klaus Aehlig and Helmut Schwichtenberg. A syntactical analysis of non-size-increasing polynomial time computation. *Submitted*, 2001. A previous version presented at LICS'00.

[Bar96] Andrew Barber. Dual intuitionistic linear logic. Technical Report ECS-LFCS-96-347, Department of Computer Science, University of Edinburgh, September 1996.

[Bib86] Wolfgang Bibel. A deductive solution for plan generation. *New Generation Computing*, 4:115–132, 1986.

[Bie94] G. Bierman. On intuitionistic linear logic. Technical Report 346, University of Cambridge, Computer Laboratory, August 1994. Revised version of PhD thesis.

[BS92] G. Bellin and P. J. Scott. On the π-calculus and linear logic. Manuscript, 1992.

[Cer95] Iliano Cervesato. Petri nets and linear logic: a case study for logic programming. In M. Alpuente and M.I. Sessa, editors, *Proceedings of the Joint Conference on Declarative Programming (GULP-PRODE'95)*, pages 313–318, Marina di Vietri, Italy, September 1995. Palladio Press.

[CF58] H. B. Curry and R. Feys. *Combinatory Logic*. North-Holland, Amsterdam, 1958.

[CHP00] Iliano Cervesato, Joshua S. Hodas, and Frank Pfenning. Efficient resource management for linear logic proof search. *Theoretical Computer Science*, 232(1–2):133–163, February 2000. Special issue on Proof Search in Type-Theoretic Languages, D. Galmiche and D. Pym, editors.

[Chu41] Alonzo Church. *The Calculi of Lambda-Conversion*. Princeton University Press, Princeton, New Jersey, 1941.

[Doš93] Kosta Došen. A historical introduction to substructural logics. In Peter Schroeder-Heister and Kosta Došen, editors, *Substructural Logics*, pages 1–30. Clarendon Press, Oxford, England, 1993.

[Gen35] Gerhard Gentzen. Untersuchungen über das logische Schließen. *Mathematische Zeitschrift*, 39:176–210, 405–431, 1935. Translated under the title *Investigations into Logical Deductions* in [Sza69].

[Gir87] J.-Y. Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.

[Gir93] J.-Y. Girard. On the unity of logic. *Annals of Pure and Applied Logic*, 59:201–217, 1993.

[GMW79] Michael J. Gordon, Robin Milner, and Christopher P. Wadsworth. *Edinburgh LCF*. Springer-Verlag LNCS 78, 1979.

[Her30] Jacques Herbrand. Recherches sur la théorie de la démonstration. *Travaux de la Société des Sciences et de Lettres de Varsovic*, 33, 1930.

[HM94] Joshua Hodas and Dale Miller. Logic programming in a fragment of intuitionistic linear logic. *Information and Computation*, 110(2):327–365, 1994. A preliminary version appeared in the Proceedings of the Sixth Annual IEEE Symposium on Logic in Computer Science, pages 32–42, Amsterdam, The Netherlands, July 1991.

[Hod94] Joshua S. Hodas. *Logic Programming in Intuitionistic Linear Logic: Theory, Design, and Implementation*. PhD thesis, University of Pennsylvania, Department of Computer and Information Science, 1994.

[Hof00a] Martin Hofmann. Linear types and non-size increasing polynomial time computation. *Theoretical Computer Science*, 2000. To appear. A previous version was presented at LICS'99.

[Hof00b] Martin Hofmann. A type system for bounded space and functional in-place update. *Nordic Journal of Computing*, November 2000. To appear. A previous version was presented as ESOP'00.

[How80] W. A. Howard. The formulae-as-types notion of construction. In J. P. Seldin and J. R. Hindley, editors, *To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, pages 479–490. Academic Press, 1980. Hitherto unpublished note of 1969, rearranged, corrected, and annotated by Howard.

[HP97] James Harland and David Pym. Resource-distribution via boolean constraints. In W. McCune, editor, *Proceedings of the 14th International Conference on Automated Deduction (CADE-14)*, pages 222–236, Townsville, Australia, July 1997. Springer-Verlag LNAI 1249.

[HT91] Kohei Honda and Mario Tokoro. An object calculus for asynchronous communication. In P. America, editor, *Proceedings of the European Conference on Object-Oriented Programming (ECOOP'91)*, pages 133–147, Geneva, Switzerland, July 1991. Springer-Verlag LNCS 512.

[Hue76] Gérard Huet. *Résolution d'équations dans des langages d'ordre* $1, 2, \ldots, \omega$. PhD thesis, Université Paris VII, September 1976.

[Kni89] Kevin Knight. Unification: A multi-disciplinary survey. *ACM Computing Surveys*, 2(1):93–124, March 1989.

[Lin92] P. Lincoln. Linear logic. *ACM SIGACT Notices*, 23(2):29–37, Spring 1992.

[Mil92] D. Miller. The π-calculus as a theory in linear logic: Preliminary results. In E. Lamma and P. Mello, editors, *Proceedings of the Workshop on Extensions of Logic Programming*, pages 242–265. Springer-Verlag LNCS 660, 1992.

[Mil99] Robin Milner. *Communicating and Mobile Systems: the π-Calculus*. Cambridge University Press, 1999.

[ML94] Per Martin-Löf. Analytic and synthetic judgements in type theory. In Paolo Parrini, editor, *Kant and Contemporary Epistemology*, pages 87–99. Kluwer Academic Publishers, 1994.

[ML96] Per Martin-Löf. On the meanings of the logical constants and the justifications of the logical laws. *Nordic Journal of Philosophical Logic*, 1(1):11–60, 1996.

[MM76] Alberto Martelli and Ugo Montanari. Unification in linear time and space: A structured presentation. Internal Report B76-16, Istituto di Elaborazione delle Informazione, Consiglio Nazionale delle Ricerche, Pisa, Italy, July 1976.

[MM82] Alberto Martelli and Ugo Montanari. An efficient unification algorithm. *ACM Transactions on Programming Languages and Systems*, 4(2):258–282, April 1982.

[MNPS91] Dale Miller, Gopalan Nadathur, Frank Pfenning, and Andre Scedrov. Uniform proofs as a foundation for logic programming. *Annals of Pure and Applied Logic*, 51:125–157, 1991.

[MOM91] N. Martí-Oliet and J. Meseguer. From Petri nets to linear logic through categories: A survey. *Journal on Foundations of Computer Science*, 2(4):297–399, December 1991.

[PD01] Frank Pfenning and Rowan Davies. A judgmental reconstruction of modal logic. *Mathematical Structures in Computer Science*, 11:511–540, 2001. Notes to an invited talk at the *Workshop on Intuitionistic Modal Logics and Applications* (IMLA'99), Trento, Italy, July 1999.

[Pol98] Robert Pollack. How to believe a machine-checked proof. In G. Sambin and J. Smith, editors, *Twenty-Five Years of Constructive Type Theory*. Oxford University Press, August 1998. Proceedings of a Congress held in Venice, Italy, October 1995.

[Pra65] Dag Prawitz. *Natural Deduction*. Almquist & Wiksell, Stockholm, 1965.

[PW78] M. S. Paterson and M. N. Wegman. Linear unification. *Journal of Computer and System Sciences*, 16(2):158–167, April 1978.

[Rob65] J. A. Robinson. A machine-oriented logic based on the resolution principle. *Journal of the ACM*, 12(1):23–41, January 1965.

[Rob71] J. A. Robinson. Computational logic: The unification computation. *Machine Intelligence*, 6:63–72, 1971.

[Sce93] A. Scedrov. A brief guide to linear logic. In G. Rozenberg and A. Salomaa, editors, *Current Trends in Theoretical Computer Science*, pages 377–394. World Scientific Publishing Company, 1993. Also in Bulletin of the European Association for Theoretical Computer Science, volume 41, pages 154–165.

[SHD93] Peter Schroeder-Heister and Kosta Došen, editors. *Substructural Logics*. Number 2 in Studies in Logic and Computation. Clarendon Press, Oxford, England, 1993.

[Sza69] M. E. Szabo, editor. *The Collected Papers of Gerhard Gentzen.* North-Holland Publishing Co., Amsterdam, 1969.

[Tro92] A. S. Troelstra. *Lectures on Linear Logic.* CSLI Lecture Notes 29, Center for the Study of Language and Information, Stanford, California, 1992.

[Tro93] A. S. Troelstra. Natural deduction for intuitionistic linear logic. Prepublication Series for Mathematical Logic and Foundations ML-93-09, Institute for Language, Logic and Computation, University of Amsterdam, 1993.

[WW01] David Walker and Kevin Watkins. On linear types and regions. In *Proceedings of the International Conference on Functional Programming (ICFP'01).* ACM Press, September 2001.