Polymorphic Linear Logic and Topos Models

R.A.G. Seely*

Abstract

We give a definition of a "linear fibration", which is a hyperdoctrine model of polymorphic linear logic, and show how to internalise the fibration, generating topos models. This gives a constructive set theoretical context for the logic of Petri nets, as recently developed by N. Martí-Oliet and J. Meseguer. Also, we sketch how this can be further extended to include the exponential operator ! . In this context, the topos model we construct can be embedded in the model constructed by A.M. Pitts.

0 Introduction

In [4], it is shown how to enrich the logic of Petri nets with *gedanken* states and processes, by embedding it into linear logic. Recently, Martí-Oliet and Meseguer have asked how to extend this even further to include polymorphism. It turns out that the process is fairly straightforward, and for maximal impact (so as to include constructive set theory), can be internalised to give topos models of polymorphic linear logic. Here we give the necessary definitions, and sketch the outline of the construction.

These notes should be thought of as a sequel to [7], extending the categorical semantics of linear logic to include polymorphism. As there, we (sketchily) describe the interpretation of polymorphic λ-calculus as the (indexed) Kleisli category induced by the !-cotriple. However, once in this context, we have another well-known model, *viz.* the internal full subcategory of a presheaf topos as constructed by A.M. Pitts [5]. Our construction does not give fullness, perhaps fortunately, but it will turn out that our model does embed faithfully into Pitts' model. (In essence his model consists of certain families, and ours corresponds to the "constant" families.)

1 Definitions

1.1 Linear fibrations

We begin with a definition:

*Research partially supported by a grant from Le Fonds F.C.A.R., Québec.

Definition 1 *A linear fibration* $(\mathbf{L}, \mathbf{S})$ *consists of*

1. *a category* $\mathbf{S}$ *with finite products, a distinguished object* U*, and exponentiation* U^A *for arbitrary* A*,*

2. *an indexed category* $\mathbf{L}$ *over* $\mathbf{S}$ *satisfying*

   (a) $Obj(\mathbf{L}(A)) = Hom_{\mathbf{S}}(A, U)$ *for all objects* A *of* $\mathbf{S}$*, and* $f^* = \mathbf{L}(f)$ *acts by composition on objects, for all morphisms* f *of* $\mathbf{S}$*,*

   (b) $\mathbf{L}(A)$ *is a linear category (i.e. ∗-autonomous with finite products—see [4, 7] for details), and* f^* *is a linear functor, for all* A, f*,*

   (c) $\mathbf{L}$ *is weakly complete: i.e. for every object* C *of* $\mathbf{S}$ *there is an (indexed) right adjoint* Π_C *to the (indexed) functor* κ_C*.*

The point of this definition is that it provides the hyperdoctrine formulation of what higher-order polymorphic linear logic is, analagous to the hyperdoctrine models defined in [6]. The usual syntactic presentation can be shown equivalent to this, *via* an equivalence-of-categories result; we leave that as an exercise.

It should also be mentioned that we have given the definition for higher-order linear logic, not merely second-order linear logic. If the reader only wants the second-order case, all that needs to be done is to drop the requirement in (1) that $\mathbf{S}$ have exponents U^A; furthermore, one may as well assume that $\mathbf{S}$ has the natural numbers as objects (where n represents U^n), or just take the base category as presented in [5]. In this case, we need only require in (2c) that Π_U exists, all other such product-functors being given by iteration in the obvious way.

1.2 Topos models

Next we define a topos model for polymorphic linear logic.

Definition 2 *A topos model of polymorphic linear logic is given by an elementary topos* $\mathbf{E}$ *together with an internal category* $\mathbf{U}$ *of* $\mathbf{E}$*. Furthermore,* $\mathbf{U}$ *must have (internally in* $\mathbf{E}$*) finite products (including the terminal object) and* A*-indexed products (for any object* A *of the sub-cartesian closed category of* $\mathbf{E}$ *generated by* U_0*—this understood internally*[1]*), and must be (internally) a ∗-autonomous category.*

Most of these notions are straightforward, and should not cause any bother. The correct notion of the involution has proved a trifle subtle (see [4]), and so we'll sketch some details here, outlining what it means to say that an internal symmetric closed monoidal category $\mathbf{U}$ has an involution $(\text{-})^{\perp} : \mathbf{U}^{op} \longrightarrow \mathbf{U}$. (The notion of the dual internal category $\mathbf{U}^{op}$ is standard.)

First, we need morphisms (of $\mathbf{E}$)

$$(\text{-})^{\perp} : U_0 \longrightarrow U_0 \quad \text{and} \quad s : U_0 \times U_0 \longrightarrow U_1$$

so that $d_0 \circ s =$ '$\multimap$' and $d_1 \circ s =$ '$\multimap$' $\circ\, (\text{-})^{\perp} \times (\text{-})^{\perp} \circ \sigma$, where σ is the "switch coordinates" isomorphism, and where '$\multimap$': $U_0 \times U_0 \longrightarrow U_0$ is the morphism giving the internal hom

[1] This is the appropriate notion for a "pure" view of the logic—if one wished to include other constants explicitly, this could be done by adding them to the generating set for the sub-ccc.

on objects. It is a straightforward exercise to mimic the construction of a contravariant functor from this data, as given in [4]. The crucial point is that one of the properties of an (internal) symmetric monoidal closed category is that there be an isomorphism $(\text{-})^\sharp : U_1 \longrightarrow U_1$ and a pullback square as follows:

$$\begin{array}{ccc} U_1 & \xrightarrow{\langle d_0, d_1\rangle} & U_0 \times U_0 \\ {\scriptstyle (\text{-})^\sharp}\downarrow & & \downarrow{\scriptstyle \langle `\mathrm{I}', `\multimap' \rangle} \\ U_1 & \xrightarrow[\langle d_0, d_1\rangle]{} & U_0 \times U_0 \end{array}$$

Next we need a pair of (inverse iso) morphisms $d : U_0 \longrightarrow U_1$ and $d^{-1} : U_0 \longrightarrow U_1$ (*sic*), (plus the appropriate equations for domains, codomains, and the identity composites), and the equations

$$s \cdot s = d^{-1}`\multimap' d : U_0 \times U_0 \longrightarrow U_1$$

$$((\text{-})^\perp \circ\ d) \cdot (d\ \circ (\text{-})^\perp) = `id'\ \circ (\text{-})^\perp : U_0 \longrightarrow U_1$$

(We leave it to the reader to decipher the notation, with the hint that $f \cdot g$ is the internal composition in $\mathbf{U}$, and $f \circ g$ is composition in $\mathbf{E}$.)

2 Constructing topos models

At this point we use a simple trick, that is part of the legacy of work done in the 1970's on indexed categories; (the details have appeared recently in [1]).

Given a linear fibration $(\mathbf{L}, \mathbf{S})$, we can construct an internal $*$-autonomous category $\mathbf{U}$ in the presheaf topos $\mathbf{E} =_{def} \mathbf{Sets}^{\mathbf{S}^{op}}$. The object of objects is $U_0 = Obj\,\mathbf{L}(\text{-}) = Hom_{\mathbf{S}}(\text{-}, U)$ and the object of morphisms is $U_1 = M\varphi\mathbf{L}(\text{-})$. The rest of the structure is defined fibrewise, in the obvious manner. For example, the internal composition $\gamma : U_2 \stackrel{\cdot}{\longrightarrow} U_1$ is the natural transformation that, at an object A of $\mathbf{S}$, sends a composable pair $\langle f, g\rangle$ (of morphisms of $\mathbf{L}(A)$) to their composite (in $\mathbf{L}(A)$). Similarly, $(\text{-})^\perp : U_0 \stackrel{\cdot}{\longrightarrow} U_0$ is the natural transformation that, at A, sends an object X to $X^\perp$ (in $\mathbf{L}(A)$).

In this model, $\mathbf{U}$ has A-indexed products for any representable object A (*i.e.* for any object of $\mathbf{S}$, *via* the Yoneda embedding).

Given an internal category, the standard way to "externalise" this, to obtain a fibration, is just to take the *Hom* functor $Hom_{\mathbf{E}}(\text{-}, \mathbf{U}) : \mathbf{E}^{op} \longrightarrow \mathbf{Cat}$. In our case, however, we can cut this back to $\mathbf{S}$ again, *via* the Yoneda embedding $H : \mathbf{S} \longrightarrow \mathbf{E}$, and so obtain a fibration $\mathbf{L}'$ over $\mathbf{S}$. Then $\mathbf{L}'(A) = Hom_{\mathbf{E}}(HA, \mathbf{U})$ is the category whose objects are natural transformations $HA \stackrel{\cdot}{\longrightarrow} HU$, or equivalently (by the Yoneda Lemma) morphisms $A \longrightarrow U$ in $\mathbf{S}$, and whose morphisms are natural transformations $HA \stackrel{\cdot}{\longrightarrow} M\varphi\mathbf{L}(\text{-})$, or equivalently, morphisms of $\mathbf{L}(A)$. Thus we see that $(\mathbf{L}, \mathbf{S})$ is equivalent to $(\mathbf{L}', \mathbf{S})$.

So we finally end up with the

Theorem 1 *Suppose* $(\mathbf{L}, \mathbf{S})$ *is a linear fibration. Then* $\mathbf{Sets}^{\mathbf{S}^{op}}$ *is a topos model of polymorphic linear logic, with internal category* $\mathbf{U}$ *as constructed above. Furthermore the "externalisation" of this internal model is, when restricted* via *the Yoneda embedding to the original base* $\mathbf{S}$, *equivalent to the original fibration.*

Remark There is another well-known method of constructing internal categories from fibrations, *viz.* the "presheaves on the Grothendieck construction" method of A.M. Pitts [5]. It is perhaps worth pointing out that that approach does not seem to work in this setting—the cartesian closed structure of the fibres of the fibration being essential to the process. If one tries to replace the cartesian structure by the monoidal structure of linear logic, one quickly finds that at several points one wants the tensor to be a real product (*e.g.* to have projections), or the unit to be a terminal. In fact, in the category $Gr(\mathbf{L})$ of "elements" of $\mathbf{L}$, one cannot even make (the canonical image of) U into (the object of objects of) an internal category. However, once we pass to the cartesian closed structure generated by the exponentials (section 3), we shall find that our model embeds into Pitts' model.

3 The exponential operator !

3.1 Girard fibrations

Of course, although the term *linear logic* ought to refer only to the logic of the additive and multiplicative connectives, it is usually used to include also the exponential operators ! and ?. In [7] it was pointed out that ! has the structure of a cotriple (*a.k.a.* comonad), and in the fibrational context (of quantification), the cotriple must be indexed in the obvious way. Since the quantification considered in [7] was not polymorphic, but only predicate, it might be worth sketching some of the details here.

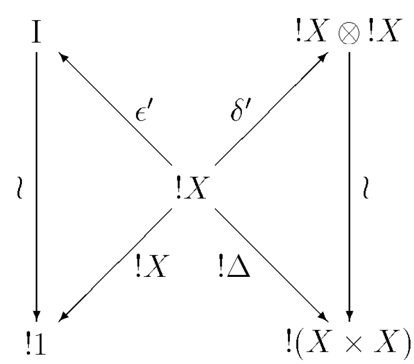
Definition 3 *A Girard fibration* $(\mathbf{G}, \mathbf{S})$ *is a linear fibration for which*

1. *each fibre* $\mathbf{G}(A)$ *is a Girard category, in the sense of [7],* via *an indexed cotriple ! , and*
2. *each "inverse image functor"* f^* *preserves this structure ("on the nose").*

To elaborate, this means that we have a cotriple $!_A : \mathbf{G}(A) \longrightarrow \mathbf{G}(A)$ (for each A), so that $!X$ carries a comonoid structure in $\mathbf{G}(A)$, for each object X of $\mathbf{G}(A)$:

$$\mathrm{I}_A \xleftarrow{\epsilon'_A(X)} !_A X \xrightarrow{\delta'_A(X)} !_A X \otimes_A !_A X$$

which is the image *via* natural (and indexed) isomorphisms $\mathrm{I}_A \xrightarrow{\sim} !_A 1_A$ and $!_A X \otimes_A !_A Y \xrightarrow{\sim} !_A(X \times_A Y)$ of the canonical cartesian comonoid structure; *i.e.* the following diagram commutes (omitting subscripts):



By saying that each f^* preserves this structure, we imply that f^* commutes with these morphisms, and in particular with !; *i.e.* $f^*(!_A X) = !_A f^*(X)$. (We ignore the question of "up to coherent iso" in these notes.)

As in [7], from a Girard fibration $(\mathbf{G}, \mathbf{S})$ we can construct the Kleisli category (indexed, of course) $(\mathbf{G}_!, \mathbf{S})$; each fibre $\mathbf{G}_!(A) = (\mathbf{G}(A))_!$. As in [7], these fibres are cartesian closed, and so we end up with a hyperdoctrine model of polymorphic λ-calculus. (The quantifier Π_C remains the same, as in [7].)

3.2 Topos models

To get the notion of topos model of the full Girard logic, one need only add to the notion of a topos model from section 1.2 the internalisation of the above notions. So we must have an internal cotriple on $\mathbf{U}$, given by an internal functor $! : \mathbf{U} \longrightarrow \mathbf{U}$ and internal natural transformations $\epsilon : ! \stackrel{\cdot}{\longrightarrow} id$ and $\delta : ! \stackrel{\cdot}{\longrightarrow} !!$, together with internal comonoid structure given by internal $\epsilon' : ! \stackrel{\cdot}{\longrightarrow}$ 'I' and $\delta' : ! \stackrel{\cdot}{\longrightarrow} ! \otimes !$ plus appropriate isos, commutative diagrams and the lot.

It is an easy exercise to see that this data allows the internal construction of the (internal) Kleisli category $\mathbf{U}_!$, and then it is easy to see that $\mathbf{U}_!$ is cartesian closed, as in [7], and is an internal topos model of polymorphic λ-calculus.

We shall further leave it to the reader to verify that the process of constructing topos models is coherent with respect to these notions: that starting with a Girard fibration, we can construct a topos model of full Girard logic, and so (*via* the Kleisli construction) a topos model of polymorphic λ-calculus, or we can arrive at this topos model by first using the Kleisli construction on the fibration.

3.3 The connection with Pitts' model

In this context, we have in front of us two topos models of polymorphic λ-caluclus, *viz.* the model $\mathbf{U}_!$ in the topos $\mathbf{E} = \mathbf{Sets}^{\mathbf{S}^{op}}$ and the model constructed by A.M. Pitts in [5], which we shall denote $\mathbf{U}'$ in the topos $\mathbf{E}' = \mathbf{Sets}^{Gr(\mathbf{G}_!)^{op}}$. (The latter is an internal full subcategory, the former is not. For the details of the construction of Pitts' model, see [5].) What is the connection between these?

First note that there is a geometric morphism between the toposes induced by the "projection" functor $p : Gr(\mathbf{G}_!) \longrightarrow \mathbf{S}$ which sends an object $(A, X : A \longrightarrow U)$ to A, and a morphism (α, f) to α. (There is another geometric morphism in the reverse direction induced by the right adjoint T to p, which embeds $\mathbf{S}$ in $Gr(\mathbf{G}_!)$ *via* terminals.) The functor p^* then carries the internal category $\mathbf{U}_!$ to an internal category (which I shall denote here by $\mathbf{U}$—the context should help avoid confusion) in $\mathbf{E}'$.

It is perhaps not too surprising that each of these internal categories (in $\mathbf{E}'$) has the same object of objects: $U_0 = U_0' = HTU$, the (representable) functor which sends $(A, X : A \longrightarrow U)$ to $Hom_{\mathbf{S}}(A, U) = Obj\,\mathbf{G}_!(A)$. However, $\mathbf{U}$ has "fewer" morphisms than (the full subcategory) $\mathbf{U}'$: U_1 sends $(A, X : A \longrightarrow U)$ to $M\varphi\mathbf{G}_!(A)$, and U_1' sends $(A, X : A \longrightarrow U)$ to $Hom_{Gr(G_!)}((A, X : A \longrightarrow U), (U^2, ` \Rightarrow ' : U^2 \longrightarrow U))$. (These may be thought of as "families" of $\mathbf{G}_!(A)$-morphisms, indexed by X. From this point of view, $U_1(X)$ consists of the "constant" families.) Recall that the morphism '$\Rightarrow$': $U^2 \longrightarrow U$ (in $\mathbf{S}$) is the composite '$\multimap$' $\circ$ ('!' $\times$ id_U); *i.e.* internally '$u \Rightarrow v$' = '$!u \multimap v$'.

Then, there is a faithful internal functor (preserving the polymorphic λ-calculus structure) $K : \mathbf{U} \longrightarrow \mathbf{U}'$, which is identity on objects:

$K_0 : U_0 \stackrel{\cdot}{\longrightarrow} U_0'$ is the identity natural transformation, and

$K_1 : U_1 \stackrel{\cdot}{\longrightarrow} U_1'$ is the natural transformation that, at an object $(A, X : A \longrightarrow U)$, sends a morphism $f : Z \longrightarrow Y$ (of $\mathbf{G}_!(A)$) to $(\langle Z, Y \rangle, `f\text{'})$, where $`f\text{'} : X \longrightarrow \top \longrightarrow (Z \Rightarrow Y)$ is the canonical "name" of f.

To check that K is a functor as claimed will be left as an exercise.

References

[1] A. Asperti and S. Martini Categorical models of polymorphism, Technical report, Dipartimento di Informatica, Università di Pisa, 1989.

[2] M. Barr $*$-*Autonomous Categories*, Springer Lecture Notes in Math. 752 (1979).

[3] J.-Y. Girard Linear Logic, Theoretical Computer Science 50 (1987), 1 - 102.

[4] N. Martí-Oliet and J. Meseguer From Petri nets to linear logic, in D. Pitt et al., eds. *Category Theory and Computer Science*, Manchester 1989, Springer Lecture Notes in Computer Science 389 (1989).

[5] A.M. Pitts Polymorphism is set-theoretic . . . constructively, in D. Pitt, ed. *Category Theory and Computer Science*, Edinburgh 1987, Springer Lecture Notes in Computer Science 283 (1988).

[6] R.A.G. Seely Categorical semantics for higher order polymorphic lambda calculus, Journal of Symbolic Logic 52 (1987) 969 - 989.

[7] R.A.G. Seely Linear logic, $*$-autonomous categories and cofree coalgebras, in J. Gray and A. Scedrov, eds., *Categories in Computer Science and Logic* (Proc. A.M.S. Summer Research Conference, June 1987), Contemporary Mathematics 92, (Am. Math. Soc. 1989).

Department of Mathematics
John Abbott College
C.P. 2000
Ste. Anne de Bellevue
Québec H9X 3L9

Department of Mathematics
McGill University
805 Sherbrooke St. W.
Montréal
Québec H3A 2K6