DIAGRAMMES

PHILIP J. SCOTT

Functorial polymorphism and semantic parametricity

Diagrammes, tome 22 (1989), p. 77-90

<http://www.numdam.org/item?id=DIA_1989__22__77_0>

© Université Paris 7, UER math., 1989, tous droits réservés.

L'accès aux archives de la revue « Diagrammes » implique l'accord avec les conditions générales d'utilisation (http://www.numdam.org/legal.php). Toute utilisation commerciale ou impression systématique est constitutive d'une infraction pénale. Toute copie ou impression de ce fichier doit contenir la présente mention de copyright.

NUMDAM

Article numérisé dans le cadre du programme
Numérisation de documents anciens mathématiques
http://www.numdam.org/

DIAGRAMMES VOLUME 22 , 1989

ACTES DES JOURNEES E.L.I.T.

(UNIV. PARIS 7, 27 JUIN-2 JUILLET 1988)

Functorial Polymorphism and Semantic Parametricity

Philip J. Scott

Department of Mathematics
University of Ottawa
Ottawa, Ont., Canada

Dans cet exposé, je décrit une interpretation functorielle du polymorphisme paramétrique, suggeré par le système *F* de Girard. Dans ce cadre, les types sont des foncteurs multivariants, les termes sont des transformations dinaturelles, et les quantificateurs sont des "ends" de foncteurs, tous définis sur une catégorie cartésienne fermé complète (au sens approprié). Les tentatives de trouver une notion de composition des transformations dinaturelles conduisent à des critères sémantiques intéressants pour la paramétricité. Je présente aussi un modèle compositionnel dans le topos Effectif.

This study was motivated by the naive question: "What mathematical structures underlie Girard's system *F* of polymorphic lambda calculus? " ([Gi 72 , 86, 87]) . In what follows I shall summarize the contents of my talk at this conference and briefly mention some more recent work. Detailed proofs, extensions, and elaborations of this material, developed jointly with Bainbridge, Freyd, and Scedrov are just currently entering the literature ([BFSSa], [BFSSb]), to which the reader is referred.

The simplest description of *F* is : the theory of proofs of intuitionistic second order propositional calculus. Thus, *types* of *F* are the formulas of intuitionist second-order propositional calculus, while *terms* denote proofs of their respective types.

Now *types* are freely generated from propositional variables $\alpha, \beta, \ldots$ by implication ($\Rightarrow$) and universal type quantification ($\forall$), while *terms* are freely generated from individual variables $x_1, \ldots, x_n, \ldots$ by two kinds of rules :

(1) First-order $\lambda\beta\eta$-rules, corresponding to the familiar introduction and elimination rules for $\Rightarrow$.

(2) Second-order $\lambda\beta\eta$-rules, corresponding to introduction and elimination rules for $\forall$:

$$\frac{t : A}{\Lambda\alpha.t : \forall\, \alpha\, A(\alpha)}\ (*) \qquad \frac{s: \forall\, \alpha\, A(\alpha) \quad \tau \text{ a type}}{s[\tau] : A(\alpha := \tau)}$$

i.e. the analog of usual lambda calculus rules, but using Λ and type-evaluation [ ] .

reçu le 19/09/1989

The first-order lambda calculus in (1) offers no problems in interpretation: if we add conjunction-types, it is essentially equivalent to the notion of cartesian closed category (cf. [LS]). The problem is the second-order part in (2): how do we interpret *variable types* , *quantification over types* , and *type evaluation* ?

My motivation for a functorial framework arose from the "polymorphic identity" term $\Lambda\alpha.\lambda x{:}\alpha.x$ of type $\forall\alpha(\alpha\Rightarrow\alpha)$ which corresponds to the obvious proof-tree. For each type A, $\Lambda\alpha(\lambda x{:}\alpha.x)[A] = \lambda x{:}A.x$ = the identity function at type A. That is, the term $\Lambda\alpha(\lambda x{:}\alpha.x)$, when applied to a type A, gives the identity function on A. Thus $\Lambda\alpha(\lambda x{:}\alpha.x)$ is a *type-indexed family of functions, uniformly the identity at each type.* In general, $\Lambda\alpha.t$ denotes a type-indexed family of "uniform" functions.

Syntactically, such terms possess all sorts of naturality. Categorical intuition suggested these uniform families should act like "natural transformations" between functors induced by (syntactic) types. So, as a first approximation to a "model", let types = functors and terms = natural transformations between types, all defined over some cartesian closed category (=ccc) $\mathcal{C}$ of "ground" types. Note that constant functors really play the role of "ground" types. Unfortunately, the function space type $\alpha \Rightarrow \beta$ should correspond to the internal hom functor $(\ )^{(\ )}$, also denoted $(\ ) \Rightarrow (\ ) : \mathcal{C}^{\circ} \times \mathcal{C} \rightarrow \mathcal{C}$ which is contravariant in its first argument and covariant in its second . Alas, the type $\alpha \Rightarrow \alpha$ would then have to be both contravariant and covariant in α , i.e. $\alpha \Rightarrow \alpha$ is just not a functor. Equally problematic is the meaning of $\forall$ as some kind of "large" internal product.

There are at least two appraches to these problems. One method, familiar from denotational semantics ([Gi 87], [CGW]), is to move to a category of retract pairs. This serves to obliterate the difference between co- and contravariance , and indeed supports (covariant) functorial models (*loc. cit.).* However, the definition of $\forall$ in this framework, as an indexed family, still permits *ad hoc* elements and so does not quite capture Strachey's notion of *parametric* polymorphism (cf. [R83]).

Independently of [Gi 87] *et al* , I (with my colleague Stewart Bainbridge) proposed to meet contravariance head-on by using a calculus of multivariant functors and appropriate "multivariant" natural transformations . This theory, developed by Yoneda[Y60] and later Dubuc and Street[DS70] (see also [Mac]) suggested setting *types* = multivariant functors $(\mathcal{C}^{\circ})^{n} \times \mathcal{C}^{n} \rightarrow \mathcal{C}$, *terms* = dinatural transformations between types, and *universal quantification* = "ends" of functors, all defined over some appropriate "internally complete" ccc $\mathcal{C}$. In keeping with categorical logic tradition, variable types are (covariant) projection functors and–curiously–the universal property of ends closely resembles Girard's problematical second-order quantification rules (at least for term models). There are difficulties:

1. Dinatural transformations do not compose, in general.
2. An end, as a kind of limit, is a more sophisticated quantifier than $\forall$. We do not get exactly the logicians' second-order $\lambda\beta\eta$-rules, but more subtle "parametric" conditions, in line with Reynolds' ideas.
3. For any of this to make sense, we still needed an appropriately complete ccc C.

Nonetheless, we obtained a rather pretty calculus with many suggestive features. It is shown in [BFSS a,b] that there is a model of this framework, by considering certain functors and dinatural transformations on ***PER*** , the category of *partial equivalence relations on the natural numbers*. ***PER*** is the externalization of an "internally complete ccc" , the Moggi-Hyland category of Modest Sets, in the Realizability Topos.

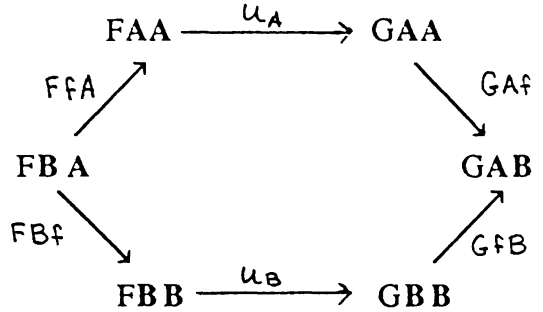
In what follows, I briefly outline the general calculus and some of the results in the references.

1. THE FUNCTORIAL CALCULUS

C is a cartesian closed category . The general plan is to interpret types as ranging over some class of multivariant n-ary functors $F: (C^{\circ})^n \times C^n \to C$ and to interpret terms as ranging over some appropriate class of "multivariant" natural transformations.

DEFINITION 1.0. A DINATURAL TRANSFORMATION between two functors $F, G: (C^{\circ})^n \times C^n \to C$ is a family of morphisms $u = \{u_A : FAA \to GAA \mid A \in C^n\}$ satisfying the following HEXAGON CONDITION: for any vector of morphisms $f: A \to B$,

(*)



In many examples u is given by a "uniform algorithm" for each object A.

Two special cases of note:

1. If F and G are actually covariant (i.e. considered dummy in their contravariant argument) then dinatural transformations coincide with natural.

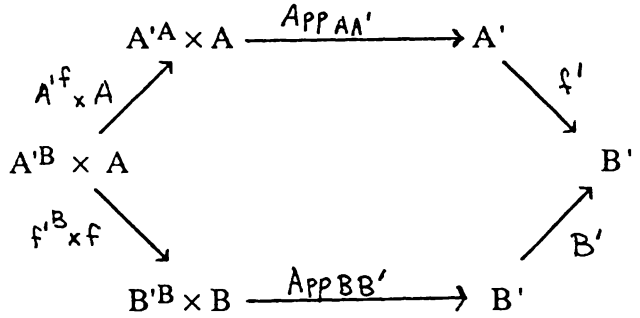
2. If F (*resp.* G) is constant, dinaturality reduces to the case of a WEDGE (*resp.* co-WEDGE) (cf. [Mac], Yoneda[Y60]).

EXAMPLES 1.1: These examples are given in SETS but make sense more generally in any cartesian closed category C. We use set-theoretic notation.

(i) "Polymorphic identity": let K_1 = the constant functor with value the terminal object **1** and $(\ )^{(\ )}$ the internal hom. Consider a wedge $u : K_1 \longrightarrow (\ )^{(\ )}$, where $u_A : \mathbf{1} \longrightarrow A^A$ is (uniformly) the "name" of the identity on A. The wedge condition reduces to the truism $f \circ id_A = id_B \circ f$.

(ii) "Uniform Church numeral **n** " : for each object A in C, the family $\mathbf{n} = \{\mathbf{n}_A : A^A \longrightarrow A^A \mid A \in C\}$ determines a dinatural transformation $(\ )^{(\ )} \longrightarrow (\ )^{(\ )}$, where $\mathbf{n}_A$ maps $h \mapsto h^n$, the n-fold composition of h with itself. The dinaturality condition says that for any $g \in B^A$, $f \circ (g \circ f)^n = (f \circ g)^n \circ f$, an instance of associativity. Again note the "uniformity" in the definition of (each component of) **n**

(iii) APPLICATION: Consider the family App , with components $App_{AA'} : (A')^A \times A \longrightarrow A'$ given by application (or evaluation) in the ccc C, for each pair of objects A, A'. Then for all $f: A \longrightarrow B$ and $f': A' \longrightarrow B'$, the following hexagon commutes:



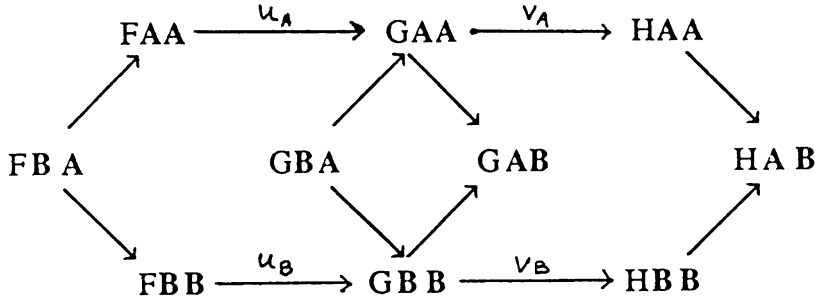
This says if $g \in A'^B$, $f'((g \circ f)(a)) = (f' \circ g)(f(a))$. App is a dinatural transformation (between functors defined below, Example 1.6).

A special case of the above example: let D be a fixed object in C. Consider the dinatural transformation $app : D^{(\ )} \times (\ ) \longrightarrow K_D$, where app_A: $D^A \times A \longrightarrow D$ is evaluation in the ccc C.

(iv) *Fixed Point combinators* : Suppose there were to exist a dinatural transformation $Y : (\ )^{(\ )} \to id$, where the *id* denotes the identity functor. The hexagon condition for the family $Y = \{Y_A : A^{A} \to A \mid A \in C\}$ means (using set-theoretic notation): For any $f \in B^A$ and $g \in A^B$, $f(Y_A(g \circ f)) = Y_B(f \circ g)$. In particular, setting $A \equiv B$ and g = the identity on A, we see Y_A *must be a fixed point combinator at each type* A. Least-fixed-point combinators in denotational semantics are examples of such [BFSSb].

The most annoying aspect of the calculus of dinatural transformations is the failure of composition. One attempts to compose two dinatural transformations $u : F \longrightarrow G$ and $v : G \longrightarrow H$ by horizontally "merging" the two hexagons; i.e. define their composite $v \circ u$ by the formula $(v \circ u)_A = v_A \circ u_A$. Then for any $f : A \longrightarrow B$, consider

(**)



While both hexagons individually commute, the outer hexagon need not commute (see [BFSSb]).

FACT 1.2: If the middle diamond in diagram (**) above is a pullback, then we *can* in fact compose the dinatural transformations u and v.

The problem of composing dinatural and other classes of generalized natural transformations has been examined by various authors (e.g. Eilenberg & Kelly [66], Kelly [72]). At a general level these problems are quite intricate and only partial solutions are known.

In section 2 below, these obstacles to compositionality will be resolved for certain large classes of multivariant functors and dinatural transformations intrinsic to the Realizability Topos. This includes at least those functors and dinatural transformations definable in second-order polymorphic lambda calculus.

We can still say a bit more about compositionality at the general level. Fix functors F, G, H : $(C^{\circ})^n \times C^n \longrightarrow C$. Let u , v denote families of

morphisms $u_A : FAA \to GAA$, $v_A : GAA \to HAA$, $A \in \mathcal{C}$, not necessarily dinatural. Let D_u be the set of f for which the family u is dinatural; that is $f \in D_u$ iff the hexagon (*) in Definition 1.0 commutes. D_v and $D_{v \circ u}$ are defined similarly.

PROPOSITION 1.3 (Vertical merging of hexagons). D_u is a subcategory.

PROPOSITION 1.4. (Horizontal merging wrt isomorphisms). Any isomorphism in D_u and D_v is necessarily in $D_{v \circ u}$.

The reason for 1.4 is that , if all maps are isos (and thus preserved by functors) then the middle diamond in (**) is a pullback. Now use Fact 1.2 .

1.5. SOME FUNCTORS

We can build new functors from old by various operations. For example, given two functors F and G

(i) Their *product* is constructed pointwise; i.e., on objects,
$(F \times G)(A,B) = F(A,B) \times G(A,B)$,

(ii) Their *twisted exponential* or *function space* G^F is the functor
$G^F(A,B) = G(A,B)^{F(B,A)}$.

Remarkably, G^F is the categorically defined exponentiation in certain models, an example of which will be described in Section 2.

The operations of product and twisted exponential of functors may be described as follows:

PRODUCTS: $F \times G$: $(\mathcal{C}^\circ)^n \times \mathcal{C}^n \to \mathcal{C}$ =

$$(\mathcal{C}^\circ)^n \times \mathcal{C}^n \xrightarrow{<F,G>} \mathcal{C} \times \mathcal{C} \xrightarrow{\times} \mathcal{C}$$

TWISTED EXPONENTIALS: G^F: $(\mathcal{C}^\circ)^n \times \mathcal{C}^n \to \mathcal{C}$ =

$$(\mathcal{C}^\circ)^n \times \mathcal{C}^n \xrightarrow{<F^*,G>} \mathcal{C}^\circ \times \mathcal{C} \xrightarrow{0^0} \mathcal{C},$$

where we write $F^* : (\mathcal{C}^\circ)^n \times \mathcal{C}^n \to \mathcal{C}^\circ$ for the composite

$$(\mathcal{C}^\circ)^n \times \mathcal{C}^n \cong \mathcal{C}^n \times (\mathcal{C}^\circ)^n \cong (\mathcal{C}^{\circ\circ})^n \times (\mathcal{C}^\circ)^n \cong ((\mathcal{C}^\circ)^n \times \mathcal{C}^n)^\circ \xrightarrow{F^\circ} \mathcal{C}^\circ \quad .$$

1.6 Example: Consider covariant projections $P_i : (\mathcal{C}^\circ)^2 \times \mathcal{C}^2 \to \mathcal{C}$, $i = 1, 2$, given by $P_i(A_1A_2 ; B_1B_2) = B_i$.

The dinat $App_{P1,P2}$: $P_2^{P1} \times P_1 \longrightarrow P_2$ has components $(App_{P1,P2})_{AA'}$; $(P_2^{P1} \times P_1)(AA', AA') \longrightarrow P_2(AA', AA')$, where $(App_{P1,P2})_{AA'} : A'^{A} \times A \longrightarrow A'$ is the application in Example 1.1 (iii) .

There is a natural bijection $\mathbf{dinat}(F \times G, H) \cong \mathbf{dinat}(F, H^G)$. We thus obtain something like a cartesian closed "noncategory" . A general theory of such ("structors") has recently been developed by Freyd[F89]. Additional monoidal structure is mentioned in [BFSSa].

Ends, due to Yoneda[Y60] (see also [Mac], pp. 218-224]), play a critical role in our semantics . Given a functor G consider a universal wedge into G: that is , an object E and a dinatural transformation $K_E \rightarrow G$, universal for all such . E, when it exists, is called the *end* of G, and is denoted by $\int_A GAA$. (This notation displays A as a bound variable; indeed, A could even be a vector of paired co- and contravariant-variable slots, some slots possibly empty). There may be other variables in the expression for G than those specifically noted; $\int_A GAA$ is functorial in these other variables.

One may think of $\int_A GAA$ as a *subset* of a "large" product $\Pi_A GAA$; namely,

$$\int_A GAA = \{g \in \Pi_A GAA \mid G(A,f)(g_A) = G(f,B)(g_B) \text{ for all } f: A \longrightarrow B \in C\} .$$

Note this product is taken over the class of all objects of *C*. In the case that G is covariant , $\int_A GAA = \lim(G)$, the limit of G .

PROPOSITION 1.7: Maps $\mathbf{1} \longrightarrow \int_A FAA \Rightarrow GAA$ are in bijective correspondence with dinatural transformations from F to G (and this correspondence is natural in any other variables).

Example 1.8: The *polymorphic natural numbers* = the interpretation of the type $\forall\alpha((\alpha \Rightarrow \alpha) \Rightarrow (\alpha \Rightarrow \alpha))$, i.e. $\int_A (A \Rightarrow A) \Rightarrow (A \Rightarrow A)$. By 1.7, its set of global elements $\cong$ **dinat** ($()^{()}$, $()^{()}$). In **SET** there is a proper class of such dinats. In the model ***PER*** discussed below, the numerals are "standard", i.e. the only dinats of $()^{()}$ to itself correspond to the Church numerals (cf. Example 1.1 (ii)).

Remark 1.9: The entire treatment of this section may be rephrased in a fibrational way, as in Seely[87], over a kind of Lawvere theory of n-ary multivariant functors.

2. PER MODELS

A *partial equivalence relation* (= per) E on a set A is a symmetric, transitive relation $E \subseteq A \times A$. E is an equivalence relation on $dom_E = \{a \in A \mid aEa\} \subseteq A$, thus partitions dom_E into disjoint classes. We consider the set *per* (N) of pers on the natural numbers N.

DEFINITION 2.1: (the category of pers) ***PER***(N) is the following category: its *objects* are per(N), the pers on N. Given $A, A' \in$ ***PER***(N), a *morphism* from A to A' is named by a partial recursive function f which induces a map of quotients $dom_A / A \longrightarrow dom_{A'} / A'$; that is: f names a morphism if, whenever nAm, then f(n), f(m) are defined and f(n)A'f(m).

Two morphisms $f,g: A \longrightarrow A'$ are *equal* if the induced maps of quotients are equal, i.e., $\forall$ m,n $\in dom_A$, nAm implies f(n), g(m) are defined and f(n) A' g(m); equivalently, f(n) A' g(n) for all $n \in dom_A$.

PROPOSITION 2.2: ***PER***(N) is a cartesian closed category.

REMARK 2.3: Definition 2.1 and Proposition 2.2 make sense if we replace N by any partial combinatory algebra $A = (A,\bullet,S,K)$ (e.g. Barendregt, [LS] , Breazu-Tannen and Coquand [87]).

From now on, we restrict ourselves to the category ***PER***(N), denoted ***PER***. However all subsequent discussions apply equally to the more general ***PER***(*A*), for *A* a combinatory algebra, as in 2.3.

A fundamental subcategory of ***PER*** is ***I***, the category whose objects are all pers (on N) but whose only maps are named by the identity function on the natural numbers N. Note that there is at most one map in ***I*** between two pers E and E', hence ***I*** can be identified with the partially ordered set of pers ordered by inclusion. It is misleading, however, to call the maps of ***I*** *inclusion* maps. They need not be monomorphisms (Note that every per is included in the maximal per; the corresponding map in ***I*** is the map that collapses an object onto **1**).

PROPOSITION 2.4: Every morphism in ***PER*** may be decomposed into an isomorphism followed by an ***I*** - map followed by an isomorphism.

DEFINITION 2.6: A *realizable* functor $F: \mathbf{PER} \longrightarrow \mathbf{PER}$ is one which which takes ***I*** to ***I*** and for which there exists a mapping Φ from the set of partial

recursive functions to itself such that for any morphism of pers from E to E' named by f, F(f) is named by $\Phi(f)$.

Almost any functor which arises in practice is realizable. One can build up a reasonable family inductively by starting from constant and projection functors (which are realizable) and close under products, twisted exponentials (see 1.6), and of course under substitution. Indeed, any functor definable in polymorphic lambda calculus is realizable (see below).

REMARK: Among the realizable functors are those functors on ***PER*** that are given internally in an important model of intuitionistic set theory and higher order logic, the Realizability Universe (or Effective Topos). As first pointed out by Moggi, this Universe contains a non-trivial, complete small cartesian closed category, the *Modest Sets.* (Carboni, et al [88], Hyland [88] , Hyland *et al* [87]. For a discussion of various notions of internal completeness satisfied by Modest Sets, cf. Robinson[89]) . Viewed externally, *Modest Sets* are equivalent to ***PER*** . This point of view was important to the approach described in this paper.

DEFINITION 2.7: Let F,G: $(\boldsymbol{PER}^{\circ})^n \times \boldsymbol{PER}^n \longrightarrow \boldsymbol{PER}$ be realizable functors. A family $u = \{u_A: FAA \longrightarrow GAA \mid A \in \boldsymbol{PER}^n\}$, *not* necessarily dinatural, is called a REALIZABLE FAMILY if there is a single partial recursive function φ such that each component u_A is named by φ .

So a realizable family $u = \{u_A: FAA \longrightarrow GAA\}$ has the property that there is a *single* numerical code n such that all components u_A *are named by the same algorithm* : the nth partial recursive function.

In the following proposition, we refer to the notation introduced before 1.3

PROPOSITION 2.8: For any realizable family u , D_u contains I .

This says: families given by a single (uniform) algorithm satisfy the hexagon property with respect to I -maps (just chase around the hexagon, with the oblique edges being inclusions of pers).

We now state the first fundamental theorem of this approach to semantics of polymorphism:

THEOREM 2.9: Realizable dinatural transformations compose.
Proof: Using the notation introduced before Proposition 1.3, suppose u and v are realizable dinaturals. The composition $v \circ u$ is of course realizable (compose

the two uniform algorithms) and therefore by Proposition 2.8 $D_{\nu \circ u}$ contains I, i.e. we can compose families dinat with respect to I-maps. By Proposition 1.4 $D_{\nu \circ u}$ contains all isomorphisms (i.e. we can *always* compose families dinat with respect to isos). Proposition 2.4 and Proposition 1.3 (vertical merge) yield the theorem.

COROLLARY 2.10: For each n, the realizable functors $(PER^{\circ})^n \times PER^n \longrightarrow PER$ and realizable dinatural transformations between them form a ccc.

Proof: That we have a category is immediate from the theorem. The ccc structure is given by products and twisted exponentials described in 1.5.

Note: we do not know if realizable families are necessarily dinatural.

We now relativize the notion of end discussed in 1.5 by restricting functors to realizable functors, and dinatural transformations to realizable transformations. Henceforth, $\int_A GAA$ will denote these REALIZABLE ENDS.

The second fundamental theorem of this approach is

THEOREM 2.11: Realizable ends exist.

The per $\int_A GAA$ is obtained by first taking the intersection of all pers GAA, then taking the subper corresponding to the dinaturality condition (see [BFSSb]).

REMARK 2.12: It is a remarkable fact that the realizable end $\int_A (A \Rightarrow A) \Rightarrow (A \Rightarrow A)$ is the per given by ordinary equality on the natural numbers N. This is true even if we substitute the intersection $\cap_A$ instead of the realizable end $\int_A$. This should be contrasted with the situation in SET (see 1.8 above).

3. SYNTAX

The dinatural calculus over PER permits a Soundness Theorem for the original syntax of Girard's F. By methods similar to those already in Girard's thesis [Gi 72], given a term $t : \tau$, *erase all types* to obtain an untyped lambda term $|t|$. We can associate with $|t|$ a gödel number e_t of a partial recursive function. Suppose, for simplicity, t has a single free variable $x:\sigma$.

THEOREM 3.1 If the free type-variables of σ, τ are contained in $\{\alpha_1, \ldots, \alpha_k\}$ then σ, τ induce realizable functors $|\sigma|, |\tau| : (PER^{\circ})^k \times PER^k \longrightarrow PER$ and e_t names a realizable dinat family $|\sigma|(A, A) \rightarrow |\tau|(A, A)$, $A \in PER$ Moreover, if $\vdash t_1 = t_2$ in system F then e_{t_1}, e_{t_2} name the same dinat.

One interest of dinatural transformations among computer scientists is that they provide a heretofore unexplored source of interesting, consistent equations in the syntax of F *(cf.* Wadler[89]).

Example 3.2 (Wadler, Breazu-Tannen, Scedrov): Add product types to F. Consider the term M: $\forall\alpha\forall\beta\ (\alpha\Rightarrow\beta) \Rightarrow ((\alpha\times \alpha)\Rightarrow(\beta \times \beta))$. Hence for any α, β we have $M[\alpha][\beta]$: $(\alpha\Rightarrow\beta) \Rightarrow ((\alpha\times \alpha)\Rightarrow(\beta \times \beta))$. Recall, from Example 1.6 that variables correspond to covariant projections $P_i : (\mathcal{C}^{\circ})^2 \times \mathcal{C}^2 \longrightarrow \mathcal{C}$, $i = 1, 2$, given by $P_i(A_1A_2 ; B_1B_2) = B_i$. Here $\mathcal{C}$ = the ccc of first-order lambda terms over quantifier-free types. Using the calculus of section 1.5, $M[\alpha][\beta]$ translates into a dinatural transformation $P_2P1 \longrightarrow (P_2 \times P_2)^{P1 \times P1}$. Dinaturality says (using Set-notation): for any types A_i , B_i , arrows f_i: $A_i \longrightarrow B_i$, and g: $B_1 \longrightarrow A_2$,

$$(f_2 \times f_2) \circ (M[A_1][A_2] (g \circ f_1)) = (M[B_1][B_2] (f_2 \circ g)) \circ (f_1 \times f_1) .$$

Special instances of this equation have been actively discussed in the literature. Note the above equation is actually *provable* in F .

In fact, there is a general result. J.-Y. Girard has pointed out that an induction technique he developed shows simply typed lambda terms yield *provable* instances of dinaturality between definable (& functorial) simple types. On the other hand, there are known instances of dinaturality which are true in ***PER***, thus consistent with F , but are not provable.

There is considerably more structure (both syntactical and semantical) in the dinatural calculus which I have no room to include here. *Some* of it is in [BFSSa] and [FGSS]. The latter reference also connects up the dinatural view with Girard's more recent work.

4. PARAMETRICITY: A BRIEF LOOK

What are the essential features of *parametric* polymorphism? Aside from uniformly given algorithms for all types, Reynolds[R83, RP88] isolated some useful semantic properties, among which we mention:

(i) parametricity as a kind of invariance under relations between type values.
(ii) the existence of initial T-algebras for definable covariant functors T .

Dinaturality itself was a first approximation to (i): the hexagon condition shows how a uniform algorithm varies along morphisms (i.e. functional relations) between type values. More generally , our attempts to find compositional classes of dinatural transformations have led to a panopoly of *relation-theoretic* versions of dinaturality which space does not permit me to go into here. The key idea is: given relations between type values, a parametric polymorphic algorithm should (uniformly) take related inputs to related outputs.

Taking this idea to an extreme, in [BFSSb] we present a relation-theoretic modelling of system *F* based on the category of "regular" relations on *PER* (i.e. regular subobjects $R \subseteq A \times A'$). The essential point (cf. Reynolds [R83]) is that we can inductively define a relational interpretation of types in which $\forall$ is not an intersection (= product) but only those elements therein which are invariant under all regular relations. This eliminates *ad hoc* elements in the product and guarantees the following theorem, illustrating theme (ii):

Theorem 4.1 : Let T be a covariant functor definable by a type expression of System *F* . Then the interpretation of $\forall\alpha((T\alpha \Rightarrow \alpha) \Rightarrow \alpha)$ is initial in the category of T-algebras on *PER* .

There are connections of initial T-algebras with various strong forms of dinaturality, cf. [Ryu89] and Freyd's theory of structors [F89].

The same techniques described in Theorem 3.1 also yield a Soundness Theorem for the relational semantics above.

BIBLIOGRAPHY

[BFSS a] Bainbridge, E.S., Freyd, P.J., Scedrov, A., Scott, P.J. Functorial polymorphism. In: Huet, G. *Logical Foundations of Functional Programming, Proceedings University of Texas Programming Institute*, Austin, Texas, June 1987, ed. by G. Huet, to appear.

[BFSS b] Bainbridge, E.S., Freyd, P.J., Scedrov, A., Scott, P.J. Functorial polymorphism, *Theoretical Computer Science,* to appear.

Barendregt, H.P. , *The Lambda Calculus (Revised Edition)*, Studies in Logic and the Foundations of Mathematics, North-Holland, 1984.

Breazu-Tannen, V., Coquand, T [87] Extensional models for polymorphism. *Proc. TAPSOFT '87 - CFLP, Pisa.* Springer LNCS 250. Expanded version to appear in *Theor. Comp. Science.*

Carboni, A., Freyd, P., Scedrov, A. [88] A categorical approach to realizability and polymorphic types. *Proc. 3rd ACM Workshop on the Mathematical Foundations of the Programming Semantics*, New Orleans, April, 1987, ed. by M. Main et al., Springer LNCS 298, 1988, pp. 23-42.

[CGW] Coquand, T., Gunter, C.A., Winskel, G. Domain theoretic models for polymorphism. *Information and Computation*, to appear.

[DS70] Dubuc, E.J., Street, R. , Dinatural transformations, *Reports of the Midwest Category Seminar IV, Springer LNM* Vol. 137 [1970], pp. 126-128.

Eilenberg, S., Kelly, G.M. A generalization of the functorial calculus, *J. Algebra* 3 [1966], pp. 366-375.

[F89] Freyd, P.J. *Structural Polymorphism I, II, III (Preliminary Report)* Univ. of Pennsylvania, Jan. 1989.

[FGSS] Freyd, P.J., Girard, J.Y., Scedrov, A., Scott, P.J. [88] Semantic parametricity in polymorphic lambda calculus. *Proc. 3rd IEEE Symposium on Logic in Computer Science*, Edinburgh, Scotland, July 1988.

[Gi 72] Girard, J.-Y. *Interprétation fonctionnelle et élimination des coupures de l'arithmétique d'ordre supérieur.* These de Doctorat d'Etat, Université de Paris VII.

[Gi 86] Girard, J.-Y. The system F of variables types, fifteen years later. *Theor. Comp. Science* 45 [1986] pp. 159-192.

[Gi87] Girard, J.-Y. Le Lambda Calcul Du Second Ordre, *Séminaire Bourbaki, Astérisque* 152-153 (1987), pp. 173-185.

Hasagawa, Ryu . *Parametric Polymorphism and Internal Representations of Recursive Type Definitions*, Manuscript, 12 April, 1989

Hyland, J.M.E. A small complete category, *Ann. Pure and Applied Logic,* V.40, (1988) 135-165.

Hyland, J.M.E., Robinson, E.P., Rosolini, G. The discrete objects in the Effective Topos. *Preprint*, 1987.

Kelly, M. Many-variable Functional Calculus I, *Coherence in Categories*, SLNM281 , S. MacLane, ed., 1977, pp. 66-105.

[LS] Lambek, J., Scott, P.J. *Introduction to higher-order categorical logic.* Cambridge University Press, Studies in Advanced Mathematics 7, 1986.

[Mac] MacLane, S. *Categories for the working Mathematician*, Graduate Texts in Mathematics 5, Springer-Verlag, 1971.

[R83] Reynolds, J.C. Types, abstraction, and parametric polymorphism. In: *Information Processing '83*, ed. by R.E.A. Mason. North-Holland, Amsterdam, pp. 513-523.

[R84] Reynolds, J.C. Polymorphism is not set-theoretic. *Symposium on Semantics of Data Types*, ed. by Kahn et al., Springer LNCS 173, 1984.

[RP88] Reynolds, J.C., Plotkin, G.D. On functors expressible in the polymorphic typed lambda calculus. To appear in *Information and Computation.*

Robinson, E.P. [89] How Complete is ***PER*** ? *Proc. 4th IEEE Symposium on Logic in Computer Science*, Asilomar, California, June 1989.

Seely, R.A.G. [87] Categorical semantics for higher-order polymorphic lambda calculus. *J. Symbolic Logic* **52** [1987] pp. 969-989.

Wadler, P. Theorems for Free! To appear in: *4th International Symposium on Functional Languages and Computer Architecture* , London, Sept. 1989.

[Y60] Yoneda, N. On Ext and exact sequences *J. Fac. Sci.*, Tokyo Sec 1, Vol. 8, 1960, pp.507-526.