CDD: 511.3

THE RULES-AS-TYPES INTERPRETATION OF SCHRÖDER-HEISTER'S EXTENSION OF NATURAL DEDUCTION

EDWARD HERMANN HAEUSLER

Department of Informatics,
Catholic University of Rio de Janeiro,
Rua Marquês de São Vicente, 225, Gávea
22453-900 RIO DE JANEIRO, RJ
BRAZIL
hermann@inf.puc-rio.br

LUIZ CARLOS P.D. PEREIRA

Department of Philosophy,
Catholic University of Rio de Janeiro,
Rua Marquês de São Vicente, 225, Gávea
22453-900 RIO DE JANEIRO, RJ
BRAZIL
luiz@fil.puc-rio.br

1. INTRODUCTION

A natural extension of Natural Deduction (N.D.) was introduced by Schroeder-Heister (Schroeder (1984)) where not only formulae but also rules could be used as hypotheses. This fact immediately allowed for the possibility of rules of arbitrary level that could discharge not only assumption-formulae but also as-

sumption-rules. This extension of N.D. was used in the definition of abstract introduction and elimination schemes for which the set of intuitionistic sentential operators $\{\bot, \rightarrow, \wedge, \vee\}$ was shown to be complete, i.e., for any sentential operator Φ, if the introduction and elimination rules for Φ are instances of the abstract schemes then, there is an intuitionistic formula F constructed out of the intuitionistic sentential operators and the sentential variables occurring in $\Phi(A_1, \ldots, A_n)$ such that $\vdash_{I+\Phi} \Phi(A_1, \ldots, A_n) \leftrightarrow F$. This idea made it possible to define an *arbitrary level (abstract) Natural Deduction* and, under certain conditions, to obtain an abstract version of normalization and strong normalization results.

The aim of the present paper is to define a typed λ-Calculus which has types of arbitrary levels (formula-as-types and rule-as-types) and to provide it with a categorical denotational semantics. It can also be shown that this typed λ-Calculus is *Curry-Howard* isomorphic to Schroeder-Heister's abstract N.D., from now on denoted by $\mathcal{HC}$.

The System λ_R is defined in the traditional way, that is, through rules for Type construction, Term construction and reduction rules.

2. TYPES

- A_1, B_1, C_1, ..., A_n, B_n, C_n, ... are types of level 0.
- If X_1, ..., X_n are types of level 0 and Φ is a type constructor, then $\Phi(X_1, \ldots, X_n)$ is a type of level 0.
- If $T_1, \ldots, T_n$ are types, B is a type of level 0, and k is the greatest level among $T_1, \ldots, T_n$, then $(T_1^\dagger, \ldots, T_n^\dagger / B)$ is a type of level $k+1$. Here $T^\dagger$ denotes the string obtained from T by replacing all the occurences of / and $\rightarrow$ by $\rightarrow$ and / respectively.

Examples :

- (A/B) is a type of level 1.
- $((A \rightarrow B), D/C)$ is a type of level 2.
- $(((A/B), D \rightarrow C)/E)$ is a type of level 3.

We need some auxiliary concepts and definitions before presenting the rules for term contruction. The funtions ρ_1 and ρ_2 are defined as:

- $\rho_1(B) = \emptyset$, if B is of level 0.
- $\rho_2(B) = \{B\}$, if B is of level 0.
- $\rho_1(T_1^\dagger, \ldots, T_n^\dagger/B) = \rho_2(T_1) \cup \ldots \cup \rho_2(T_n)$
- $\rho_2(T_1^\dagger, \ldots, T_n^\dagger/B) = \{B\}$

A *context* is a sequence (sometimes viewed as a set) of the form :

$$x_1 : T_1; x_2 : T_2; \ldots; x_n : T_n$$

We will use capital Greek letters Δ, Θ and Γ to denote contexts.

A *pattern* is simply a variable x associated to a type expression T. The level of a pattern is the level of its type. Patterns of level 0 correspond to formula assumptions in traditional N.D. Patterns, in general, will correspond to assumption rules.

Before we define the class of terms, let us consider the following example of a deduction scheme in N.D. style:

$$\frac{\begin{array}{ccc}[H_1^1], \ldots, [H_1^{j_1}] & & [H_n^1], \ldots, [H_n^{j_n}] \\ \vdots & \cdots & \vdots \\ G_1 & & G_n\end{array}}{X}$$

We can relate each hypothesis $H_i^{j_p}$ either to a term $x : H_i^{j_p}[x : H_i^{j_p}]$ or to a pattern of the corresponding type. Thus, the term related to the deduction of the corresponding G_p has free occurences of x. The term corresponding to the deduction of the whole type must then have x as a bound variable, since the type corresponding to x is discharged by the rule. Well, we may think of a more computational kind of rule where the discharging mechanism is applied only to the deduction of the appropriate premiss. Thus our λ-Calculus should take into account that the binding of a variable in a term representing a rule, or better, an application of it, is only related to the term representing the deduction of the corresponding premiss. This selective way of binding variables is then represented by a list (of lists) of variables where each member of the list is related, in the correspondent ordering, to the term where these variables are bound. This is considered in the second item of the following definition.

- $x : X[x : X]$ is a term for any type X of level 0 and any variable x. Notice that a pattern is a distinguished kind of term.
- If $t_1 : T_1[\Theta_1], \ldots, t_n : T_n[\Theta_n]$ are terms and x is a pattern of type $T_1^\dagger, \ldots, T_n^\dagger/X$, then

$$[\vec{x}_1 \mid \ldots \mid \vec{x}_n] < x, t_1, \ldots, t_n >: X[x : T_1^\dagger, \ldots, T_n^\dagger/X, \Theta_1 - [\vec{x}_1], \ldots, \Theta_n - [\vec{x}_n]]$$

is a term where, $\vec{x}_i$ is a list of variables whose respective types are in $\rho_1(T_i)$.

We can represent the introduction and elimination rules for an arbitrary constant Φ in our λ-Calculus as following. Let Φ be a constant which has n introduction schemes, each of them similar to the rule shown at the beginning of this subsection.

Its n schemes are, each of them, represented by the following abstraction terms Λ^i_Φ, $i = 1, \ldots, n$.

Let $\vec{x^i_j} : \vec{T^i_j}$ and $t^i_j(\vec{x^i_j}) : G^i_j[\Gamma^i_j]$ for $0 \leq j \leq \rho_i$. Then

$$\Lambda^i_\Phi[\vec{x^i_1}, \ldots, \vec{x^i_{\rho_i}}](t^i_1, \ldots, t^i_{\rho_i}) : \Phi(F_1, \ldots, F_n)[\Gamma^i_1, \ldots, \Gamma^i_{\rho_i}]$$

is a term.

Related to these n introduction schemes we have the following elimination scheme:

Let

- $\vec{y^i_i} \equiv [y^i_1 : (T^i_{1,1})^\dagger, \ldots, (T^i_{1,m_1})^\dagger / G^i_1, \cdots, y^i_{\rho_i} : (T^i_{\rho_i,1})^\dagger, \ldots, (T_{\rho_i,m^i_{\rho_i}})^\dagger / G^i_{\rho_i}]$
- $v_i : A[\Delta_i]$
- $v : \Phi(T^\dagger_1, \ldots, T^\dagger_n)[\Delta]$

then,

$App_\Phi[\vec{y_1}, \ldots, \vec{y_n}] < v, v_1, \ldots, v_n >: A[\Delta, \Delta_i - \vec{y_i}]$ is a term.

Fact 1 *It is worth noticing that from the way we construct our terms for representing introduction and elimination rules, we have the following:*

> *For each variable (either free or bounded) of type rule R occurring in a term of the form $\Lambda^i_\Phi[\vec{x^i_1}, \ldots, \vec{x^i_{\rho_i}}]$ $(t^i_1, \ldots, t^i_{\rho_i}) : \Phi(F_1, \ldots, F_n)[\Gamma^i_1, \ldots, \Gamma^i_{\rho_i}]$ and for any term t that is formed by an application of Φ-Elimination, there is a subterm of t of type $\rho_2(R)$ with variables occurrences with types from the set $\rho_1(R)$, among others.*

This fact plays a major role in the reduction rules shown below. Whenever there is no risk of confusion we will omit the index (i) of Λ^i_Φ.

3. OPERATIONAL SEMANTICS

A term is called a *redex* if it has one of the following forms:

$$App_\Phi[\vec{y_1}, \ldots, \vec{y_n}] < \Lambda_\Phi[\vec{x_1^i}, \ldots, \vec{x_{\rho_.}^i}](t_1^i, \ldots, t_{\rho_i}^i), v_1, \ldots, v_n >: A[\Gamma]$$

or

$$App_\Phi[\Delta] < App_\Psi[\Theta] < t_1, \ldots, t_k >, v_1, \ldots, v_n >: A[\Gamma]$$

The former is called an *operational redex*, while the latter is called a *structural redex* and plays the rôle of permutative reductions in N.D.(cf. (Prawitz (1965))).

Before we define reduction rules for these redexes, we will take a brief look at what happens in an N.D. framework. For the sake of simplicity, we will write down the reduction rule with Λ_Φ representing the first introduction rule for Φ (in the ordering of the minor premisses in the elimination rule). Let Δ_i be the following context:

$$x_1 : (\Theta_1^i)^\dagger / G_1^i, \ldots, x_{\mu_1} : (\Theta_{\mu_1}^i)^\dagger / G_{\mu_1}^i$$

Notice that each variable x_k, in this context, is of type $((\Theta_1^i) \rightarrow G_1^i)^\dagger$. This is the type of the terms that correspond to the i-th premiss in the (first) Φ introduction rule. Fact 1 says that we must have in the first Φ introduction rule the set of types discharged (variables bounded to the new term) is Θ_1^1 for the first premiss, Θ_2^1 for the second, and so on. Let's consider the following redex (associated with this first introduction rule).

$$\mathbf{App}_\Phi[\emptyset \mid \Delta_1 \mid \ldots \mid \Delta_n](\Lambda_\Phi[\Theta_2^1 \mid \ldots \Theta_k^1](t'_1, \ldots, t'_k), d_1, \ldots, d_n) : A[\Gamma]$$

Consider the process of filling all of the x_j (of type $(\Theta_j^1)^\dagger/G_j^1$) with the terms t'_j (of type G_j^1 and context Θ_1^j). The reduction of this redex is the result of applying this process of filling (which is a generalization of substitution) to d_1. We shall now detail this filling process of variables (representing rules) with terms (representing the associated deductions). This will be called *iterated substitution* and is illustrated below in the context of N.D.

To replace a variable x of type $\Theta^\dagger/G$ by a term u of type G and context Θ, Γ in a term $t(< x, d_1, \ldots, d_n >)$ (here we explicitly show the occurrence of x) is equivalent in N.D. to replacing a derived rule by the deduction that justifies it.

Let Π be the following derivation:

$$\dfrac{\begin{array}{c}\Pi_1 \quad\quad \Pi_n \\ H_1 \quad \cdots \quad H_n\end{array}}{\begin{array}{c}\alpha \\ \Sigma \\ \beta\end{array}}\ \mathrm{R}$$

Given a deduction Π^*

$$\begin{array}{c} H_1 \ldots H_n \\ \Pi^* \\ \alpha \end{array}$$

we can replace the application of the rule R:

$$\dfrac{H_1 \ldots H_n}{\alpha}$$

by Π^*, obtaining in this way the following derivation:

$$\begin{array}{c}\Pi_1 \quad\quad \Pi_n \\ H_1 \quad \cdots \quad H_n \\ \Pi^* \\ \alpha \\ \Sigma \\ \beta\end{array}$$

In λ_R, the process of substitution discussed above can be formalized in the following definition of iterated substitution.

Definition 3.1 Let T_i be a type of level k and B a type of level 0, then:

- $\rho_1(B) = \emptyset$ and $\rho_2(B) = \{B\}$.
- $\rho_2(T_1^\dagger, \ldots, T_n^\dagger/B) = \{B\}$.
- $\rho_1(T_1^\dagger, \ldots, T_n^\dagger/B) = \rho_2(T_1) \cup \ldots \cup \rho_2(T_n)$.
- If $d(x) : C[x : T, \Delta]$ is a term, and $t_1 : \rho_1(T)[\Gamma, \vec{y} : \rho_2(T)^\dagger]$, then the iterated substitution of x by t_1, denoted $d(x \stackrel{+}{\leftarrow} t_1) : C[\Delta, \Gamma]$ is such that:
  - If T is a formula, then $d(x \stackrel{+}{\leftarrow} t_1)$ is $d(t_1) : C[\Gamma]$.
  - If $T = H_1^1, \ldots, H_1^{j_1} \rightarrow F_1, \ldots, H_n^1, \ldots, H_n^{j_n} \rightarrow F_n/G$, then t_1 and d must have the forms:

$$d(< x, d_1, \ldots, d_n >) : C[x : T, \Delta]$$

with $d(\vec{y_i} : F_i[\Delta, \vec{y_i} : \vec{H_i}]$, and

$$t_1(x_1, \ldots, x_n) : G[x_i : \vec{H_i}^\dagger / F_i, \Gamma]$$

$$d(x \stackrel{+}{\leftarrow} t_1) : C[x : T, \Gamma, \Delta] = d(t_1(x_i \stackrel{+}{\leftarrow} d_i)) : C[\Delta, \Gamma]$$

We should consider what happens if some of the x_j are of the type of a rule in the example above when we are treating the case of introductions and eliminations. In this case, if x_j, for some j, is of the type of a rule, it must be of type $\Delta^\dagger/H_j$ where Δ is the context of d_j, in the elimination rule, which in turn is of type H_j (Fact 1).

Thus, given the following operational redex

$$App_\Phi[\vec{y_1},\ldots,\vec{y_k}] < \Lambda_\Phi[\vec{x_1^\omega},\ldots,\vec{x_{n_\omega}^\omega}](t_1^\omega,\ldots,t_{n_\omega}^\omega), v_1,\ldots,v_k >: A[\Gamma],$$

for $1 \leq \omega \leq k$, its contractum is

$$v_\omega(y_j^\omega \stackrel{+}{\leftarrow} t_j^\omega) : A[\Gamma]$$

In the same way, given the structural redex

$$App_\Phi[\Delta] < App_\Psi[\Theta] < t_1,\ldots,t_k >, v_1,\ldots,v_n >: A[\Gamma],$$

its contractum is

$$App_\Psi[\Theta](t_1, App_\Phi[\Delta](t_2, v_2,\ldots,v_n),\ldots, App_\Phi[\Delta](t_k, v_2,\ldots,v_n))$$

We use the notation $t \rhd t'$ to denote that t reduces t' in one step. $\rhd^*$ is the transitive-reflexive closure of $\rhd$.

We show below a short example of a reduction for the case where x_j is of the type of a rule in N.D. in our calculus. Consider the following deduction which has $\Phi(F)$ as a maximum formula.

$$\dfrac{\dfrac{\dfrac{\Pi_1}{A}}{\dfrac{B}{\dfrac{\Pi_2}{C}}}\qquad \dfrac{\dfrac{[A]}{\dfrac{\Pi_3}{B}}}{\dfrac{C}{\dfrac{\Pi_4}{D}}}}{\dfrac{\Phi(F)}{D}}$$

This reduces to

$$\begin{array}{c}\Pi_1\\ A\\ \Pi_3\\ B\\ \Pi_2\\ C\\ \Pi_4\\ D\end{array}$$

In λ_R the reduction is as follows.

$$App_\Phi[z] < \Lambda_\Phi[x]u([] < x, t >), v([y] < z, t'(y) >) >:$$

$$D[] \quad \rhd^* v(z \leftarrow u(x \leftarrow t'(y \leftarrow t))),$$

where $\Lambda_\Phi[x]u([] < x, t >) : \Phi(F)[]$ represents Π_2, t represents Π_1,$t'(y) : B[y : A]$ represents Π_3, and $v([y] < z, t'(y) >) : D[z : A \to B/C]$ represents Π_4.

From the relationship between Schröder-Heister N.D. ($\mathcal{HC}$) and λ_R as shown here we have the following results.

Theorem 3.1 For any proof Π of α with assumption set Γ, there is a λ_R-term $t : \alpha[\Gamma]$. For any λ_R-term $t : \alpha[\Gamma]$ there is a proof Π of α with assumption set $\Gamma' \subseteq \Gamma$.

Theorem 3.2 For any term $t : T[\Gamma]$ there is a term without redexes (Normal) $t' : T[\Gamma']$ with $\Gamma' \subseteq \Gamma$.

Theorem 3.3 Let $t(\vec{x}) : \alpha[\vec{x} : \Gamma]$ be the term associated with a proof Π in $\mathcal{HC}$ of α from Γ. Then if $t = t_1, \ldots, t_n = t'$, with t' *normal*, is a *sequence* of reductions, then the *normal proof* associated to Π can be obtained from the inverse image of t' and the *reduction sequence* in $\mathcal{HC}$ is the inverse image of $t_1, \ldots, t_n$.

$$\begin{array}{ccccccc} t_1 & \rhd & \ldots & t_i & \ldots & \rhd & t_n \\ \updownarrow & & & \updownarrow & & & \updownarrow \\ \Pi_{t_1} & \rhd & \ldots & \Pi_{t_i} & \ldots & \rhd & \Pi_{t_n} \end{array}$$

4. CATEGORICAL SEMANTICS

The categorical semantics for λ_R is defined in the usual way, interpreting Types as objects and terms as morphisms. Given a Cartesian closed category **C** with finite co-products, the denotation function [[.]] takes as arguments types, patterns and terms, and yields values in **C** (either objects or morphisms).

Types

- If T is a basic type of level 0, then $[[T]]$ is an object of C.
- If $T_1^\dagger, \ldots, T_n^\dagger / Y$ is a type of level $n > 0$, then

$$[[T_1^\dagger, \ldots, T_n^\dagger / Y]] = [[Y]]^{[[T_1]] \times, \ldots, \times [[T_n]]} \in Obj(\mathbf{C})$$

- If $\Phi(F_1, \ldots, F_n)$ is a type of level 0 and has $R_1, \ldots, R_n$ as the types of its abstraction (introduction) rules , then

$$[[\Phi(F_1, \ldots, F_n)]] = [[R_1]] + \ldots + [[R_n]]$$

If $\Gamma = x_1 : T_1, \ldots, x_n : T_n$ is a context, then we use $[[\Gamma]]$ to denote $\prod_{i=1,n}[[T_i]]$, that is, the product of the denotations of each type in the context.

Patterns

- $[[x : R[x : R]]] = I_{[[R]]}$

Terms

A term $t : T[\Gamma]$ will be interpreted as a morphism from $[[\Gamma]]$ into $[[T]]$.

- $[[x : X[x : X]]] = I_{[[X]]}$
- Given that $x : T_1^\dagger, \ldots, T_n^\dagger / Y[x : T_1^\dagger, \ldots, T_n^\dagger / Y]$ is a pattern and $t_i : Y_i[\Gamma_i \cup \Theta_i]$ are terms, $0 < i < n + 1$, provided that $T_i = \Theta_i / Y_i$ and that

$$[[t_i : Y_i[\Gamma_i \cup \Theta_i]]] = f_i : [[\Gamma_i]] \times [[\Theta_i]] \longrightarrow [[Y_i]],$$

by the property of exponentials that gives a natural isomorphim between $Hom_{\mathbf{C}}(a \times b, c)$ and $Hom_{\mathbf{C}}(a, c^b)$, we

have a unique morphism $\widehat{f}_i$ from $[[\Gamma_i]]$ into $[[Y_i]]^{[[\Theta_i]]}$. Thus, $< \widehat{f}_1, \ldots, \widehat{f}_n >$ is a morphism from $[[\Gamma]]$ into

$$[[Y_1]]^{[[\Theta_1]]} \times \ldots \times [[Y_n]]^{[[\Theta_n]]}.$$

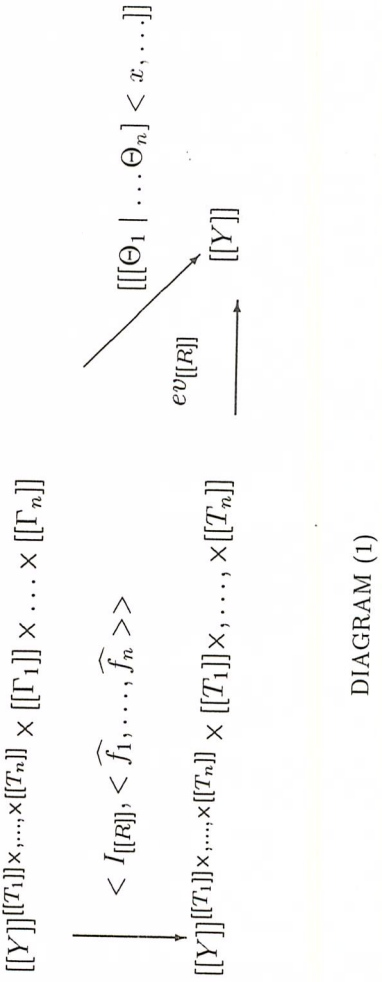

DIAGRAM (1)

Finally, we define

$$[[[\Theta_1 \mid \ldots \Theta_n] < x, t_1, \ldots, t_n >: Y[x : T_1^\dagger, \ldots, T_n^\dagger / Y, \Gamma_1, \ldots, \Gamma_n]]]$$

as $ev_{[[R]]} \circ < I_{[[R]]}, < \widehat{f_1}, \ldots, \widehat{f_n} >>$, where R is $T_1^\dagger, \ldots, T_n^\dagger / Y$ and $ev_{[[R]]}$ is the corresponding "eval" morphism of its denotation, which is of course an exponential. That the denotation given above is indeed a morphism from $[[x : R, \Gamma_1, \ldots, \Gamma_n]]$ into $[[Y]]$ is represented by the Diagram (1).

- Given that $t_i : Y_i[\Gamma_i \cup \Theta_i]$ is a term, $0 < i < n+1$, and that

$$[[t_i : Y_i[\Gamma_i \cup \Theta_i]]] = f_i : [[\Gamma_i]] \times [[\Theta_i]] \longrightarrow [[Y_i]],$$

$< \widehat{f_1}, \ldots, \widehat{f_n} >$ is a morphism from $[[\Gamma]]$ into

$$[[Y_1]]^{[[\Theta_1]]} \times \ldots \times [[Y_n]]^{[[\Theta_n]]}$$

Thus, we define

$$[[\Lambda_\Phi[\Theta_1 \mid \ldots \mid \Theta_n] < t_1, \ldots, t_n >: Y[\Gamma_1, \ldots, \Gamma_n]]]$$

as $i_{[[R_i]]} \circ < \widehat{f_1}, \ldots, \widehat{f_n} >$, where R_i is

$$\Theta_1 \to Y_1, \ldots, \Theta_n \to Y_n / \Phi(F_1, \ldots, F_n),$$

the i-th introduction rule for Φ. So, $i_{[[R_i]]}$ is the injection morphism given by the denotation of co-product of type $\Phi(F_1, \ldots, F_n)$.

- Finally we have the morphism associated with the elimination rules. Let
  - $\vec{y_i} \equiv [y_1^i : T_{1,1}^i, \ldots, T_{1,m_1}^i / G_1^i, \cdots, y_{\rho_i}^i : T_{\rho_i,1}^i, \ldots, T_{\rho_i,m_{\rho_i}^i}^i / G_{\rho_i}^i]$
  - $v_i : A[\Delta_i]$

– $v : \Phi(F_1, \ldots, F_n)[\Delta]$

Given that

$$[[\Delta]] \xrightarrow{[[v]]} [[\Phi(F_1, \ldots, F_n)]]$$

and

$$[[\vec{y_i}]] \times [[\Delta_i']] \xrightarrow{[[v_i]]} [[A]]$$

where $\Delta_i' = \Delta_i - \vec{y_i}$, let $DISTRB = ev \circ dist$ be the morphism that plays the role of the distributivity law of the product with regard to the co-product. Thus, observing that

$$[[\Phi]] = [[\vec{y_1}]] + \ldots + [[y_n]],$$

we finally define

$$[[App_\Phi[\vec{y_1}, \ldots, \vec{y_n}] < v, v_1, \ldots, v_n >: A[\Delta, \Delta_1', \ldots, \Delta_n']]]$$

as the composition

$$[[[v_1]] \circ < I_{[[\vec{y_1}]]}, \pi_{[[\Delta_1']]} >, \ldots, [[v_n]] \circ < I_{[[\vec{y_n}]]}, \pi_{[[\Delta_n']]} >]$$

$$\circ DISTRB \circ < [[v]], I_{([[\Delta_1']] \times \ldots \times [[\Delta_n']])} >$$

By defining a suitable concept of substitution at the semantical level, i.e., a kind of a iterated composition, we can prove the adequacy of our categorial model.

Theorem 4.1 Let $t : Y[\Delta]$ be a λ_R-term such that there is $t' : Y[\Delta]$ and $t \triangleright^* t'$. Then, we have that $[[t : Y[\Delta]]] = [[t' : Y[\Delta]]]$.

5. CONCLUSION

From this theorem we can conclude that any Cartesian Closed Category with finite co-products is a model of λ_R. The addition

of types of arbitrary levels does not interfere with the basic semantical intuitions. The enlarged expressive power of λ_R relies, for example, on the possibility of considering certain assumption-formation processes as the specification of programming modules (as in MODULA-II (Wirth (1985))): the modules *hide* their implementations, but specify the interface (types of the premises, of the discharged hypothesis and of the conclusion) that they ought to have with the world.

REFERENCES

CHI, W.H. (1991). *Esquemas Abstratos para Dedução Natural, Cálculo de Sequentes e λ-Calculus Tipado.* Dissertação de Mestrado, Departamento de Informática, PUC-Rio.

HAEUSLER, E.H. & PEREIRA, L.C.P.D. (1992). "A Denotational Semantics for Arbitrary Level Typed λ-Calculus", *Monografias em Ciência da Computação* (Departamento de Informática, PUC-RIO).

POUBEL, H.W. & PEREIRA, L.C.P.D. (1993). "A Categorical Approach to Higher-Level Introduction and Elimination Rules", *Reports in Mathematical Logic* 27, pp. 3-19.

PRAWITZ, D. (1965). *Natural Deduction. A Proof Theoretical Study* (Stockholm, Almqvist-Wiksell).

SCHRÖDER-HEISTER, P. (1984). "A Natural Extension of Natural Deduction", *Journal of Symbolic Logic*, vol. 49.

WIRTH, N. (1985). *Programming in Modula-2* (Berlin, Springer-Verlag).