Institute for Language, Logic and Information

INTENSIONAL LAMBEK CALCULI

THEORY AND APPLICATION

Andreja Prijatelj

ITLI Prepublication Series
for Logic, Semantics and Philosophy of Language LP-89-06

University of Amsterdam

Instituut voor Taal, Logica en Informatie
Institute for Language, Logic and Information

Faculteit der Wiskunde en Informatica
(Department of Mathematics and Computer Science)
Plantage Muidergracht 24
1018TV Amsterdam

Faculteit der Wijsbegeerte
(Department of Philosophy)
Nieuwe Doelenstraat 15
1012CP Amsterdam

INTENSIONAL LAMBEK CALCULI

THEORY AND APPLICATION

Andreja Prijatelj
Electrotechnic Faculty
University of Ljubljana

Received October 1989

C O N T E N T S

1. I N T R O D U C T I O N

The main motivation for the present paper has arisen as a challenge to extend the extensional Lambek calculus developed by van Benthem [2] to an intensional version in a natural way. Actually, three intensional deductive systems are presented below distinguished with respect to the number of structural rules they contain.
To continue with a brief outline of this paper.
Its main topic is:
meta-properties of intensional Lambek calculi.
The immediate topic of application is:
an analysis of Montague s meaning postulates by a Lambek-like mechanism in such a way as to show them superfluous.
One of the most important aims is:
a better insight into phenomena of intensionalization.
Finally, let us state the main technical results of this paper:
A cut elimination theorem for ILP_{ce} is proved.
A decision procedure for ILP_{ce} is established, being also valid for both weaker systems.
An auxiliary one-sorted system AS is constructed.
Borrowing Gallin´s idea [5] an AS -typed semantics is "translated" in such a way as to become a proper fragment of Ty_2.
A correspondence between AS and IL_c is established.
A strategy for dynamically assigning an adequate interpretation to expressions of any given category is proposed and verified.

2. INTENSIONAL LAMBEK CALCULI

2.1. Deductive systems: ILP_{ce}, IL_{ce}, IL_c

We shall start with a presentation of the deductive system ILP_{ce} of sequents: $a_1,a_2,..,a_n \rightarrow b$ where $a_1,a_2,...,a_n,b$ denote arbitrary *two-sorted simple types* inductively defined as follows:

DEFINITION (2.1.1.):
Let e,t,s, be any distinct objects, none of which is an ordered pair. The set of two-sorted simple types, called Types, is the smallest set T satisfying:
(i) e,t,s, $\in$ T
(ii) if a,b $\in$ T, then (a,b) $\in$ T

Remark:
In what follows the types defined by (i) and (ii) will be referred to as basic types and functional types respectively.
Next the forms of *axioms* and *inference rules* for the system ILP_{ce} are stated below:

axioms:
(1) $a \rightarrow a$

operational rules:
introduction of a functional type in the succedent of a sequent

(2) $\dfrac{a,T \rightarrow b}{T \rightarrow (a,b)}$ (2′) $\dfrac{T,a \rightarrow b}{T \rightarrow (a,b)}$

introduction of a functional type in the antecedent of a sequent

(3) $\dfrac{T \rightarrow a \quad U,b,V \rightarrow d}{U,T,(a,b),V \rightarrow d}$ (3′) $\dfrac{T \rightarrow a \quad U,b,V \rightarrow d}{U,(a,b),T,V \rightarrow d}$

s t r u c t u r a l   r u l e s:

permutation of antecedent types:

(4)
$$\frac{U,a,b,V \to d}{U,b,a,V \to d}$$

contraction *restricted* *to the* *basic* *type s*:

(5) $\frac{U,s,R,s,V \to a}{U,s,R,V \to a}$ (5´) $\frac{U,s,R,s,V \to a}{U,R,s,V \to a}$

expansion *restricted* *to the* *basic* *type t*:

(6) $\frac{U,R,t,V \to a}{U,t,R,t,V \to a}$ (6´) $\frac{U,t,R,V \to a}{U,t,R,t,V \to a}$

Adding finally the *cut rule*:

$$\frac{T \to a \quad U,a,V \to b}{U,T,V \to b}$$

In the above rules of inference U , V , T , R denote finite sequences of types, with the restriction that T is non-empty.

Remark:

By a distinct notation for the structural rules in ILP_{ce} their *full* or *restricted* use within the system is indicated. Further on, it will be seen that the *restriction* of *contraction* and *expansion* to the *basic types s* and *t* respectively calls for some specific tools when establishing some of the system´s meta-properties, filling the gap between the well-known strategies for the weaker and the stronger systems (i.e. L, LP and LPC_e, intuitionistic logic respectively).

Next, some comments on the *present exposition* of the system ILP_{ce} will be given. Clearly, the rules (2´), (3´), (5´) and (6´) are superfluous. Namely, the corresponding symmetric variant of any specific inference rule above is easily derivable by successive applications of the permutation rule to the appropriate premise sequent or conclusion sequent. Moreover, there are other possible equivalent formulations of the system ILP_{ce} depending on the degree of permutation that is already built into a specific inference rule itself. Thus, for example, the rules (5) and (6) could simply be replaced by:

$$\frac{s,s,V \longrightarrow a}{s,V \longrightarrow a} \quad \text{and} \quad \frac{t,V \longrightarrow a}{t,t,V \longrightarrow a}$$

respectively. However, the reason for adopting and further analyzing the ILP_{ce} version given above is closely related to the section (6). There IL_c, the present symmetric system without *permutation*, *expansion* and *cut*, is applied to the linguistic model proposed by Montague [12]. Furthermore, the introduction of the rule of *restricted expansion* into our system has been motivated by the author´s conjecture that the system IL_{ce}, i.e. the above symmetric system lacking *permutation* and *cut*, can be of use for simplifying the dynamic intensional systems recently developped by Groenendijk and Stokhof [7]. Thus the properties of the system IL_{ce} will be inspected simultaneously with those of IL_c and ILP_{ce}.
Finally, the addition of the *cut rule* to the system ILP_{ce} will be justified in the next subsection by pointing out that the number of ILP_{ce} theorems is not increased by making use of the cut rule in any ILP_{ce} derivation. However, the latter fact does not hold for the two symmetric systems IL_{ce} and IL_c; the lack of the permutation rule will turn out to be essential.

2.2. Eliminability of Cut

First of all, a standard version of the *cut-elimination theorem* [6] suitably adopted for the system ILP_{ce} will be proven. For that purpose some additional prerequisites are needed: in particular, the following notions of *functionality degree of types*, *the grade of a cut* and *the trace of a specified occurrence of a type within a given proof tree.*

<u>DEFINITION (2.2.1.)</u>:

Let f: Types $\rightarrow \mathbb{N}$ be a function, such that:

$$f(e) = 0 \ , \ f(t) = 0 \ , \ f(s) = 0 \ ,$$
$$f((a,b)) = f(a) + f(b) + 1 \ , \text{ then}$$

the f-image of any given type is precisely the functionality degree of that type.

<u>DEFINITION (2.2.2.)</u>:

The grade of the cut : $\dfrac{T \rightarrow a \qquad U,a,V \rightarrow b}{U,T,V \rightarrow b}$

is f(a), where f is defined by (2.2.1.) and a is the active type of the cut.

Finally, the trace of a specified type occurrence of a given sequent within a particular cut-free proof tree of the same sequent is defined inductively on the *size of that proof tree* (i.e. the number of sequents occurring in it).

<u>DEFINITION (2.2.3.)</u>:

Let $a_1,a_2,\ldots,a_{p-1} \rightarrow a_p$ be an endsequent of a particular cut-free ILP_{ce} proof tree of size n, denoted by Δ.
For any i {1,2,...,p} the trace of a_i, $tr_\Delta(a_i)$, within Δ is determined as follows:

(I) n = 1; Δ: $b \rightarrow b$, then $tr_\Delta(a_i) = \emptyset$, for a_i denoting either of occurrences of b in the axiom.

(II) n > 1; Here, several cases are to be considered, distinguished with respect to the last derivational step of Δ.

i) rule (2):

$$\frac{\overset{\Delta'}{\overbrace{b, Z \to c}}}{Z \to (b,c)}, \text{ then}$$

$$tr_{\Delta}(a_i) = \begin{cases} 0, & \text{if } a_i \text{ is the ind. occ. of } (b,c); \\ tr_{\Delta'}(a_i)+1, & \text{otherwise;} \end{cases}$$

ii) rule (3):

$$\frac{\overset{\Delta'}{\overbrace{Z \to b}} \quad \overset{\Delta''}{\overbrace{U,c,V \to d}}}{U,Z,(b,c),V \to d}, \text{ then}$$

$$tr_{\Delta}(a_i) = \begin{cases} 0, & \text{if } a_i \text{ is the ind. occ. of } (b,c); \\ tr_{\Delta'}(a_i)+1, & \text{if } a_i \text{ occurs in } Z; \\ tr_{\Delta''}(a_i)+1, & \text{otherwise;} \end{cases}$$

iii) permutation rule:

$$\frac{\overset{\Delta'}{\overbrace{U,b,c,V \to d}}}{U,c,b,V \to d}, \text{ then}$$

$tr_{\Delta}(a_i) = tr_{\Delta'}(a_i)+1$, no matter which type occurrence in $U,c,b,V \to d$ is denoted by a_i.

iv) contraction restricted to the basic type s:

$$\frac{\overset{\Delta'}{\overbrace{U,s,R,s,V \to b}}}{U,s,R,V \to b}, \text{ then}$$

$$tr_{\Delta}(a_i) = \begin{cases} 0, & \text{if } a_i \text{ is the ind. occ. of } s; \\ tr_{\Delta'}(a_i)+1, & \text{otherwise;} \end{cases}$$

v) expansion restricted to the basic type t:

$$\frac{\overset{\Delta'}{\overbrace{U,R,t,V \to b}}}{U,t,R,t,V \to b}, \text{ then}$$

$$tr(a_i) = \begin{cases} 0, & \text{if } a_i \text{ is any of ind. occ. of } t; \\ tr(a_i)+1, & \text{otherwise;} \end{cases}$$

Remarks:

Here "ind. occ." is a shortened form of "indicated occurrence(s)". Note, moreover, that the trace of basic types s and t can be *blocked* by *contraction* and *expansion* of relevant types respectively.

Let us now present:

P R O P O S I T I O N (2.2.4.):

Any given ILP_{ce} proof tree of a sequent can be converted to a cut-free ILP_{ce} proof tree of the same sequent. The only inference rules applied in it are the ones applied in the given proof tree with additional applications of the permutation rule in some cases.

P r o o f:

By induction on the number of applications of the cut rule in the given proof tree, provided that the following lemma is proved first.

L E M M A (2.2.5.):

For any given ILP_{ce} cut-free proof trees of two sequents having the form $T \longrightarrow a$ and $U,a,V \longrightarrow b$ respectively an ILP_{ce} cut-free proof tree of the sequent $U,T,V \longrightarrow b$ can be found. The only inference rules applied in it are the ones applied in the two given proof trees with additional applications of the permutation rule in some cases.

P r o o f:

The sequents $T \longrightarrow a$ and $U,a,V \longrightarrow b$ with the cut-free ILP_{ce} proof trees Δ_l for the former and Δ_r for the latter sequent are given by the assumption of the lemma. Then the proof tree of the sequent $U,T,V \longrightarrow b$ exists with the single application of the cut:

$$(*)\qquad \frac{\begin{array}{cc} \backslash\ \Delta_1\ / & \backslash\ \Delta_r\ / \\ T \to a & U,a,V \to b \end{array}}{U,T,V \to b}$$

as the final derivational step.
However, it will be shown that the same sequent also has a cut-free ILP_{ce} proof tree.
The proof will be given by the following nested induction: a course-of-values induction on the grade of cut (def.2.2.2), such that within each of its steps an induction on the trace of the active type of a cut with respect to a given proof tree is used. In case of a proof tree having an application of a cut as its final derivational step, and no other cut over it, definition (2.2.3.) can be used for "tracing" the active type within each specific proof tree of the left and right cut-premise respectively. The trace of the active type a of a cut, denoted by Ctr(a), is then expressible by the sum of the corresponding left and right counterparts.
Hence, for the cut given above (*), the following holds:

$$Ctr(a) = tr_{\Delta_1}(a) + tr_{\Delta_r}(a).$$

Now, it only remains to show how this single application of the cut can be shifted up the given proof tree and finally becomes superfluous.
Actually each of the *conversion steps* laid out below is either a basis or an induction step of the nested induction, so that a verification is made for the former and a proper induction hypothesis is used for the latter one.

(I) grade of the given cut is zero: f(a) = 0, or equivalently a is a basic type.

(1) Ctr(a) = 0 with respect to the given proof tree (*).
Hence, $tr_{\Delta_1}(a) = tr_{\Delta_r}(a) = 0$. It suffices to see that by def. (2.2.3.) Δ_1 can only be: a $\to$ a, and thus the application of the cut is superfluous.

(2) Ctr(a) > 0 with respect to the given proof tree (*).
Hence, the following two possibilities, splitting further into specific subcases, are to be handled:

(A) $tr_1(a) > 1$,

$$\dfrac{\dfrac{T'' \to b_1 \qquad b_2,T' \to a}{T'',(b_1,b_2),T' \to a}\;(3) \qquad U,a,V \to d}{U,T'',(b_1,b_2)T',V \to d}\;(\text{cut})$$

has to be converted into:

$$\dfrac{T'' \to b_1 \qquad \dfrac{b_2,T' \to a \qquad U,a,V \to d}{U,b_2,T',V \to d}\;(\text{cut})}{U,T'',(b_1,b_2),T',V \to d}\;(3)$$

(iii)

$$\dfrac{\dfrac{T',c,d,T'' \to a}{T',d,c,T'' \to a}\;(P) \qquad U,a,V \to b}{U,T',d,c,T'',V \to b}\;(\text{cut})$$

has to be converted into:

$$\dfrac{\dfrac{T',c,d,T'' \to a \qquad U,a,V \to b}{U,T',c,d,T'',V \to b}\;(\text{cut})}{U,T',d,c,T'',V \to b}\;(P)$$

iv)

$$\dfrac{\dfrac{T',s,R,s,T'' \to a}{T',s,R,T'' \to a}\;(\text{contr}_s) \qquad U,a,V \to b}{U,T',s,R,T'',V \to b}\;(\text{cut})$$

is to be converted into:

$$\dfrac{\dfrac{T',s,R,s,T'' \to a \qquad U,a,V \to b}{U,T',s,R,s,T'',V \to b}\;(\text{cut})}{U,T',s,R,T'',V \to b}\;(\text{contr}_s)$$

v)

$$\dfrac{\dfrac{T''',T'',t,T' \to a}{T''',t,T'',t,T' \to a}\;(\text{exp}_t) \qquad U,a,V \to b}{U,T''',t,T'',t,T',V \to b}\;(\text{cut})$$

is to be converted into:

$$\dfrac{\dfrac{T''',T'',t,T' \to a \qquad U,a,V \to b}{U,T''',T'',t,T',V \to b}\;(\text{cut})}{U,T''',t,T'',t,T',V \to b}\;(\text{exp}_t)$$

Clearly the order of application of each of the above treated ILP_{ce} inference rules and cut has been reversed, with the endsequent remaining the same while the trace of a has been decreased by one.
Thus, the hypothesis of induction on the trace of a with respect to a given proof tree can be used.
(B) $tr^{A}_{r}(a) > 0$,
Here, the cases where the last derivational step of $U,a,V \to b$ has been an application of any of the rules (2) to (6), in such a way that the condition stated in (B) is fulfilled, should be treated analogously as those of (A) above. That is, the order of application of the last inference rule in the derivation of $U,a,V \to b$ and cut with $T \to a$ is to be reversed.
However, the special case concerning the application of permutation must still be worked out:

$$\dfrac{T \to a \qquad \dfrac{U,a,d,V \to b}{U,d,a,V \to b}\,(P)}{U,d,T,V \to b}\,(cut)$$

is to be converted into:

$$\dfrac{\dfrac{T \to a \qquad U,a,d,V \to b}{U,T,d,V \to b}\,(cut)}{U,d,T,V \to b}\,(\Pi_k)$$

where Π_k denotes k successive applications of the rule (P) in case a sequence of k types is represented by T.
Again, by each of the proposed conversions the trace of a has been decreased by one. And, thus the hypothesis of induction on the trace of a becomes available.
(II) the grade of the given cut is greater than zero: $f(a) > 0$, or equivalently a is a functional type..
(1) $Ctr(a) = 0$ with respect to the given proof tree (*).
Hence, $tr_{l}(a) = tr^{A}_{r}(a) = 0$. Clearly the following possibilities occur for the left and right proof tree in question:

Δ_l is either an axiom or an application of rule (2) or (2′) is its final derivational step;
Δ_r is either an axiom or a relevant application of rule (3) or (3′) is its final derivational step;
In case one of the derivations is an axiom the application of the cut is superfluous. Hence, still four possibilities of combining the last derivational step of Δ_l with the one of Δ_r remain. However, only two of them will be handled here. The cases with a symmetric variant of each inference rule under consideration are to be treated analogously with their symmetric counterpart given below:

(i)
$$\frac{\dfrac{a_1,T \to a_2}{T \to (a_1,a_2)}\ (2) \qquad \dfrac{U' \to a_1 \quad U'',a_2,V \to b}{U,(a_1,a_2),V \to b}\ (3)}{U,T,V \to b}\ (\mathrm{cut})$$

with $a = (a_1,a_2)$ and $U = U'',U'$

is to be converted into:

$$\frac{U' \to a_1 \qquad \dfrac{a_1,T \to a_2 \quad U'',\ a_2,V \to b}{U'',a_1,T,V \to b}\ (\mathrm{cut}_2)}{U,T,V \to b}\ (\mathrm{cut}_1)$$

(ii)
$$\frac{\dfrac{a_1,T \to a_2}{T \to (a_1,a_2)}\ (2) \qquad \dfrac{V' \to a_1 \quad U,a_2,V'' \to b}{U,(a_1a_2),V',V'' \to b}\ (3')}{U,T,V',V'' \to b}\ (\mathrm{cut})$$

has to be converted into:

$$\cfrac{\cfrac{V' \to a_1 \qquad \dfrac{a_1,T \to a_2 \quad U,a_2,V'' \to b}{U,a_1,T,V'' \to b}\ (\mathrm{cut}_2)}{U,V',T,V'' \to b}\ (\mathrm{cut}_1)}{U,T,V',V'' \to b}\ (\Pi_{l.k})$$

where $\Pi_{l.k}$ denotes l.k successive applications of the rule (P) in case V′ and T represent sequences of k types and l types respectively.
By each of the conversions proposed above the grade of the cut has been decreased, since $f(a_i) < f((a_1,a_2))$ for

$i \in \{1,2\}$. Thus, the hypothesis of induction on the grade of cut can be applied to cut_2, since each of its premises by assumption satisfies the condition of the lemma, its proof tree is cut-free. Clearly after cut_2 has been eliminated the same holds true for cut_1 and thus the above induction hypothesis can be used for it as well.

(2) $Ctr(a) > 0$ with respect to the given proof tree (Δ). Further, (A) and (B) dealing with the subcases of $tr_{\Delta_1}(a) > 0$ and $tr_{\Delta_r}(a) > 0$ are analyzed in the same way as (I.2:A and B) above.

And that completes the proof of the lemma.

Q.E.D.

Remarks:

Note that by def. (2.2.3.) the condition: $tr_{\Delta_r}(a) = 0$, stated in (I.1.) above, allows the following additional possibilities for Δ_r besides its being an axiom:

(i) the last derivational step of Δ_r can also be *contraction of relevant s types;*

(ii) the last derivational step in Δ_r can also be *expansion of a relevant t type;*

However, the verification of (I.1) is not dependent on the form of Δ_r, as pointed out in the above proof.

Note also, that the cases demanding *additional applications* of the *permutation rule*, treated above, indicate that cut elimination in ILP_{ce} may *increase* the size of an initially given proof. Actually, a rather rough polynomial estimate is given below.

O b s e r v a t i o n (2.2.6.):

If $s(\Delta) = n$ for some $n \in \mathbb{N}$, then $s(\Delta') < n^3$, with $s(\Delta)$ and $s(\Delta')$ denoting the size of any given ILP_{ce} proof tree Δ and its cut-free counterpart Δ' respectively.

P r o o f:

Suppose $s(\ ) = n$. The central property to be used here is introduced first. For that purpose let $a_1, a_2, \ldots, a_m \in b$

denote an arbitrary sequent occurring in Δ. That the inequality: $m \leq n$, holds can be proved by a course-of-values induction on the size of Δ. The estimation stated above is now available as follows. Take any derivational step of Δ such that it has been furnished by some additional applications of the permutation rule in the course of converting Δ to Δ'. Using the above stated property, the number of these additional applications within any conversion step is strictly less than n^2. Finally, taking the estimation for each derivational step of Δ the claim of the observation is justified.

<u>Remark</u>:

By the above result, ILP_{ce} again gets the position between the system L and intuitionistic logic. Namely, in the process of cut elimination the size of the proofs within the former system is not increased [9] while there may be an exponential blow-up in size of the proofs within the latter system [11]. Still, there is an interesting fact to be observed. If only the permutation rule of ILP_{ce} would be replaced by a permutation in its broader sense then the same result as for L would hold for such a system.

In what follows, it will be shown that cut is *not* a *derivable rule* in the system IL_{ce} or IL_c. To put it differently, the rule cut can not be added to IL_{ce} and IL_c if the set of previously derivable theorems (that is without cut) is to be preserved. That fact will be justified by the following simple example:

$$
\dfrac{
\dfrac{
\dfrac{
s \to s \qquad \dfrac{e \to e \qquad t \to t \quad (3)}{e,(e,t) \to t} \quad (3)
}{
s,(s,e),(e,t) \to t
} \quad (2)
}{
(s,e),(e,t) \to (s,t)
}
\qquad
\dfrac{s \to s \qquad t \to t \quad (3')}{(s,t),s \to t} \quad (\mathrm{cut})
}{
(s,e),(e,t),s \to t
}
$$

Clearly, the last sequent could not be derived without the *cut rule* in any of the above mentioned systems *lacking permutation*. That fact will fully be justifiable in the next subsection where the decision procedure for the relevant systems is introduced. However, an interesting question can be raised here concerning the connection of the cut rule with the permutation rule within the system ILP_{ce}. The answer is put forward by the following

__*O b s e r v a t i o n* (2.2.7.)__:
The system ILP_{ce} is equivalent to the system IL_{ce} extended by the cut rule.

__P r o o f:__
(i) ($\leftarrow$)
By lemma (2.2.5.) cut is a derivable rule of ILP_{ce}.
(ii) ($\rightarrow$)
To prove the converse, it only remains to be shown that the permutation rule:

$$\frac{a_1,..,a_i,a_{i+1},..,a_n \to b}{a_1,..,a_{i+1},a_i,..,a_n \to b} \quad (P)$$

is deducible from the IL_{ce} inference rules using also cut. Suppose $a_1,..,a_i,a_{i+1},..,a_n \to b$ is a derivable sequent within the system under consideration. Then so is the sequent $a_i \to (a_{i+1},..,(a_n,(a_{i-1},..,(a_2,(a_1,b))..))..)$. Take the latter as the left-hand side premise of the cut by which $a_1,..,a_{i+1},a_i,..,a_n \to b$, i.e. the conclusion sequent of the rule (P) above, can be inferred. Then the right-hand side premise of cut may only be the sequent below:

$a_1,..,a_{i-1},a_{i+1},(a_{i+1},..,(a_n,(a_{i-1},..,(a_1,b)..))..),a_{i+2},..,a_n \to b$,

which clearly is derivable within the given system.
And that completes the present proof.

2.3. D e c i d a b i l i t y

After the cut-eliminability question for ILP_{ce} has been answered affirmatively, it becomes plausible to search for a decison procedure for that sequential calculus. However, neither of the already existing decision procedures for stronger Gentzen's systems as compared to ILP_{ce} can be obviously adopted here. Following too faithfully a decision procedure for the intuitionistic logic discovered by Gentzen himself [7], would not be very useful, because it depends essentially on the presence of all structural rules. The same holds for the implicational fragment of relevance logic introduced by Kripke [1[illegible]], where the presence of *full contraction* calls for such a strong tool as the well-known Kripke's lemma [4]. Therefore, we propose a simpler method designed especially for this case.

In the following discussion concerning the decidability of ILP_{ce} an *important lemma* will be introduced playing the central role in a decision procedure itself. Namely, any path of a complete proof search tree of a given sequent is to be stopped if the sequent obtained by one of the *operational rules* or *s-type contraction*, looked at in the reverse order, does not satisfy the condition stated by the lemma. Moreover, the same lemma guarantees that within each path of a complete proof search tree only a finite number of applications of s-type contraction, again looked at in the reverse order, can be performed.

Further, since the cut rule was proved to be superfluous in the system ILP_{ce} one can confine oneself to cut free derivations without loss of generality. Thus the system ILP_{ce} without cut will be used through-out the subsequent discussion concerning the decision procedure.

Let us now present the lemma by which a *necessary condition* for a sequent to occur in an ILP_{ce} proof tree of some specified sequent is expressed.

For that purpose the following notation will be introduced.

<u>Notation</u>:

Suppose Γ is any given sequent. Let then $n_b(\Gamma)$ and $n_f(\Gamma)$

denote the number of *occurrences* of the *basic type s* in Γ and the number of *occurrences* of *s* in all *functional types* of Γ respectively.

L E M M A (2.3.1.):

For every sequent Γ that occurs in an arbitrary ILP_{ce} proof tree of some specified sequent the following holds: either $n_b(\Gamma) \leq n_f(\Gamma)$, or Γ is the axiom $s \rightarrow s$.

P r o o f:

By course-of-values induction on the size of a proof tree of a given sequent.

1) The basis step is a verification whether the above stated property holds for a single axiom proof tree: either $n_b(a \rightarrow a) \leq 0$ or Γ has the form $s \rightarrow s$.
2) The induction step reduces to checking whether the same property is preserved by each inference rule supposedly being the last derivational step of a given proof tree:

(a) introduction of a functional type in the succedent of a sequent:

$$\frac{a, Z \rightarrow b}{Z \rightarrow (a,b)} \quad \begin{matrix} \Gamma_0 \\ \Gamma_1 \end{matrix}$$

Clearly $n_b(\Gamma_1) = n_b(\Gamma_0)$ and $n_f(\Gamma_1) \geq n_f(\Gamma_0)$.
Thus, using the induction hypothesis for the premises the following inequality holds:
$n_b(\Gamma_1) = n_b(\Gamma_0) \leq n_f(\Gamma_0) \leq n_f(\Gamma_1)$.

(b) introduction of a functional type in the antecedent of a sequent:

$$\frac{Z \rightarrow a \qquad U,b,V \rightarrow d}{U,Z,(a,b),V \rightarrow d}$$

Here four possibilities are to be checked, distinguished with respect to the fact that both, one or none of the premises of the given inference rule are instantiations of the axiom $s \rightarrow s$. As an example, one of the possible cases is displayed below:

$$\frac{s \rightarrow s \qquad U,b,V \rightarrow d}{U,s,(s,b),V \rightarrow d} \quad \begin{matrix} \Gamma_0, \Gamma_1 \\ \Gamma \end{matrix}$$

either:

(i) b is s and thus:

$n_b(\Gamma) = n_b(\Gamma_1) \leq n_f(\Gamma_1) < n_f(\Gamma_1) + 2 = n_f(\Gamma)$.

or:

(ii) $n_b(\Gamma) = n_b(\Gamma_1) + 1$, but also $n_f(\Gamma) = n_f(\Gamma_1) + 1$, etc. Clearly the induction hypothesis : $n_b(\Gamma_1) \leq n_f(\Gamma_1)$ was used in (i) as well as in (ii).

c) permutation of antecedent types:

$$\frac{U,a,b,V \to d}{U,b,a,V \to d} \quad \begin{matrix}\Gamma_o\\ \Gamma_1\end{matrix}$$

clearly $n_b(\Gamma_1) = n_b(\Gamma_o) \leq n_f(\Gamma_o) = n_f(\Gamma_1)$.

d) contraction restricted to the basic type s:

$$\frac{U,s,R,s,V \to a}{U,s,R,V \to a} \quad \begin{matrix}\Gamma_o\\ \Gamma_1\end{matrix}$$

the induction hypothesis implies:

$n_b(\Gamma_1) = n_b(\Gamma_o) - 1 \leq n_b(\Gamma_o) \leq n_f(\Gamma_o) \leq n_f(\Gamma_1)$.

e) expansion restricted to the basic type t:

$$\frac{U,R,t,V \to a}{U,t,R,t,V \to a} \quad \begin{matrix}\Gamma_o\\ \Gamma_1\end{matrix}$$

with the same justification as in (c) above.

Q. E. D.

Remark:

Obviously the lemma is also valid for the cut-free systems IL_{ce} and IL_c.

In what follows a complete ILP_{ce} proof search tree of any given sequent will play a central role. In order to show that a *special construction* of such a tree is finite and thus that ILP_{ce} is *decidable* a definition introducing the *functionality degree of a sequent* is presented below:

D E F I N I T I O N (2.3.2.):

Let Γ be any given sequent of the form: $a_1,a_2,\ldots,a_n \to b$, then $F(\Gamma) = f(a_1) + f(a_2) +..+ f(a_n) + f(b)$, with f defined by (2.2.1.), is the functionality degree of Γ.

***

We shall continue with:

P R O P O S I T I O N (2.3.3.):

The system ILP_{ce} is decidable.

P r o o f:

Suppose that Γ is an arbitrary sequent. *A procedure for constructing a complete proof search tree of* Γ, such that its finiteness can easily be seen, is determined below:

(i) *all distinct permutations* (P) *of the antecedent types of* Γ *with the exception of identity are performed first;*

(ii) *all possible applications of both operational rules and s-type contraction supposedly being the last derivational step of* Γ *are done next. However, each of the resulting successor nodes* Γ' *must still be checked out in the following way:*
in case $n_b(\Gamma') > n_f(\Gamma')$, *while* Γ' *is not* $s \to s$, *the path containing* Γ' *must be stopped. Clearly, by lemma (2.3.1) that path has only been a failed attempt to find a proof tree of* Γ.

(iii) *finally, all possible applications of t-type expansion looked at in the reverse order are performed.*

However, a permutation (P) will in the present proof freely be taken in its broader sense, thus collecting several successive applications of the previously given ILP_{ce} permutation rule into a single derivational step.

The important thing in further *construction* of any *path* within a complete proof search tree is that *successive applications* of the *permutation* (P) are *prohibited.*

Consider the contrary case, where clearly several successive steps of permutation result in another specific permutation by which either one of the direct successors of Γ or Γ itself is produced again.

Hence, *two sorts* of *nodes* must be distinguished within such a tree as regards to their being *obtained by permutation* in the last derivational step or by some *other* ILP_{ce} *inference rule* looked at in the reverse order. As to the former nodes one simply ignores part (i) of the strategy above, as to the latter ones (i) as well as (ii) and (iii) must be performed, clearly both with respect to some *specified node* of a tree. To conclude that the system ILP_{ce} is decidable it only remains to be seen that an ILP_{ce} complete proof search tree of any given sequent constructed as prescribed above is finite. Thus by the well-known König's lemma it suffices to verify that the following two properties are satisfied by the proof tree under consideration:

(a) *finite fork property*

Holds trivially.

(b) *finite branch (path) property*

Suppose Γ_a is an arbitrary node of a complete proof search tree of Γ and let Γ_a' be any direct successor of Γ_a. By a straightforward verification the following essential feature of such a tree becomes available:

(*) $F(\Gamma_a') \leq F(\Gamma_a)$, *with F being defined by (2.3.2.).*

Moreover, the strict inequality holds only in the case when the connection of Γ_a with Γ_a' is performed by one of the operational rules. *And* $F(\Gamma_a') = F(\Gamma_a)$ *otherwise.*

(i) Making use of (*) an *upper bound* for the number of applications of *operational rules* within an *entire* proof tree of Γ is $F(\Gamma)$. To put it differently, there may be at most $F(\Gamma)$ functional type eliminations from the antecedent or from the succedent of sequents that occur in any proof tree of Γ and hence a fortiori in each path of a complete proof search tree of Γ.

(ii) Obviously, only finitely many applications of t-type expansion looked at in the reverse order can be performed in in each path of a complete proof search tree of Γ.

(iii) By the above construction of a complete proof search tree of Γ successive applications of permutation within any of its paths are prohibited.

Finally, by the prescribed construction of the tree and (i), (ii) and (iii) above it follows that there are particular nodes of the tree below which no application of either operational rules or t-type expansion occurs within a relevant path. But then, due to the chosen tree construction only the following two possibilities remain for such a path:

either all of its nodes are connected by s-type contraction or by permutation and s-type contraction in succession.
However, by the condition of the lemma (2.3.1.) which has been built into the procedure itself any such path will be stopped after finitely many steps. More precisely, leting Γ_a be such a node $n_f(\Gamma_a)-n_b(\Gamma_a)$ is the number of applications of s-type contraction, looked at in the reverse order, for the former as well as for the latter path.
Hence, every path ends up after a finite number of steps, either by an axiom, or because none of the rules can or may further be applied. And that completes the present proof. Thus a decision procedure for the ILP_{ce} system has been established.

Q.E.D.

Remark:
Obviously, the decidability of the systems IL_{ce} and IL_c is shown by the above proof as well.

3. AUXILIARY SEQUENTIAL CALCULUS AS

In this section we shall introduce an auxiliary deductive system AS of sequents: $a_1,\ldots,a_n \to b$ where $a_1,\ldots.a_n,b$ denote arbitrary elements of Types defined by (2.1.1.). Actually, the basic type s itself *never* occurs in an AS derivable sequent due to the restriction for the AS axiom scheme given below. And just by the latter fact a kind of *one-sorted* construction of a given auxiliary system is indicated, providing for a natural transition to Montague intensional systems [12], discussed in section (6).
In what follows a presentation of the *deductive system AS* will be given:

a x i o m s:

(1) $a \to a$, if a is not s;

r u l e s o f i n f e r e n c e:

introduction of a functional type in the succedent of a sequent

(2) $\dfrac{a,T \to b}{T \to (a,b)}$ (2′) $\dfrac{T,a \to b}{T \to (a,b)}$

introduction of a functional type in the antecedent of a sequent:

(3) $\dfrac{T \to a \quad U,b,V \to d}{U,T,(a,b),V \to d}$ (3′) $\dfrac{T \to a \quad U,b,V \to d}{U,(a,b),T,V \to d}$

intensionalization of a type in the succedent of a sequent:

(4) $$\dfrac{T \to a}{T \to (s,a)}$$

constrained intensionalization of a type in the antecedent of a sequent:

(5) $$\frac{U,a,V \rightarrow b}{U,(s,a),V \rightarrow b}$$

provided that the occurrence of a type (s,a) produced by an application of (5) is transmitted to the succedent of a sequent by an application of either (2) or (2') within a proof tree of a given sequent.

In the above inference rules U,V,T are any finite sequences of types with the restriction that T is non-empty.

Remark:

Note that (1), (2) and (3) above actually have the same form as their respective IL_c counterparts. However, the restriction in AS (1) is inherited also by the next two AS inference rules. Thus *no operation* with a *basic type s* can be performed within the auxiliary system. Still *intensional liftings* of arbitrary types can be made by rule (4) above. An analogy with the role of *cap operator* in Montague´s one-sorted approach can thus easily be seen. Finally, by AS rule (5) a *restricted* effect of Montague´s *cup operator* is achieved. That will become clear, further on, when the corresponding translation rule will be stated. However, note for example, that (s,e) $\rightarrow$ e is not an AS derivable sequent due to the constraint in (5) above.

O b s e r v a t i o n (3.1):

The deductive system AS is decidable.

P r o o f:

For each AS inference rule the following holds:

$F(\Gamma_p) < F(\Gamma_c)$, in case of one-premise inference rule and

$F(\Gamma_{p_1}) + F(\Gamma_{p_2}) < F(\Gamma_c)$, in case of two-premise inference rule, with F as in definition (2.3.2) and Γ_p, Γ_{p_1}, Γ_{p_2} and Γ_c denoting premise sequents and conclusion sequent respectively.

Clearly, by each of the above inference rules, if looked at in the reverse order, a special kind of *functional type* *elimination* is defined. But the number of all possible functional type eliminations in any AS proof tree of some specified sequent ⊢ is limited by F( ). And hence, using some of the arguments given in the proof of the proposition (2.3.3.) that are relevant for the system AS, it is easy to see that a complete proof search tree of is finite. And that completes the present proof.

4. L A M B D A - T Y P E D   S E M A N T I C S

4.1. G e n e r a l   F r a m e

In what follows a semantic apparatus based on λ-typed terms assigning the meaning to any given AS or IL_c derivation of a sequent will be introduced. Note that a change of terminology is present above: " derivation " will from now on usually stand for " proof tree ", the latter term being somewhat characteristic for purely syntactic contexts.
In this subsection a *general frame* of the lambda-typed semantics *common* to both systems will be given.

To start with the set of *λ-typed terms* Λ inductively defined as follows:

<u>DEFINITION (4.1.1.)</u>:

(1) Firstly the *primitive symbols* are determined:

(i) For each a, element of Types, there is a denumerable list of *variables*: $v_a, v'_a, \ldots$

(ii) For each a, element of Types, there is a denumerable list of *constants*: $c'_a, c_a, \ldots$

(iii) The *improper* symbols are: the abstractor   and parentheses (,).

(2) A recursive *characterization* of the set Λ_a of λ-typed terms of an arbitrary a, element of Types, will be given next:

(i) $x_a \in \Lambda_a$

(ii) $c_a \in \Lambda_a$

(iii) If $t_{(a,b)} \in \Lambda_{(a,b)}$ and $t_a \in \Lambda_a$, then $t_{(a,b)}(t_a) \in \Lambda_b$

(iv) If $t_b \in \Lambda_b$, then $\lambda x_a t_b \in \Lambda_{(a,b)}$

(v) Λ_a is the smallest set satisfying clauses (i) to (iv);

(3) The set of all λ-typed terms Λ is $U\{\Lambda_a;\ a \in \text{Types}\}$.

<u>Remark</u>:

Throughout (1), (2) and (3) above the set Types is given by definition (2.1.1.) with the following *intended interpretation:*

Let D and I be any non-empty sets. The *denotational domains* D_a are defined by induction on $a \in$ Types as follows:

$D_e = D$, $D_t = \{0,1\}$, $D_s = I$ and $D_{(a,b)} = D_b^{D_a}$, the latter representing all set theoretic functions from D_a to D_b.

To continue with:

<u>Notation</u>:

(i) x_a, y_a, z_a...denote arbitrary variables of type a.

(ii) c_a, d_a, e_a...denote arbitrary constants of type a.

(iii) t_a, t'_a, t''_a,... denote arbitrary terms of type a.

(iv) $\equiv$ denotes syntactic equality between λ-typed terms.

(v) $t_a[x_b:=t_b]$ and $t_a[c_b:=t_b]$ denote the substitution of a λ-typed term t_b for x_b and c_b respectively in a given λ-typed term t_a, where a and b are any elements of Types.

Moreover, the following convention will respected in subsequent sections: λ-typed terms are considered to be *syntactically equal* if they are *convertible* into each other by means of *α-conversion*, as defined in Barendregt [1]:

<u>Convention (4.1.2.)</u>:

Let t_a, t'_a. Then $t_a \equiv t'_a$, provided that t'_a results from t_a by a series of changes of bound variables in t_a resulting from the replacement of a part $\lambda x_b t_c$ of t_a by $\lambda y_b(t_c[x_b:=y_b])$, where y_b does *not* occur in t_c.

So far, the main prerequisites for an exposition of the lambda-typed semantics have been worked out.

4.2. Type Driven Translations with respect to AS and IL_c

In this subsection a *method* for assigning a *type-driven translation* to a particular AS or ILC_c *derivation* of some specified sequent will be presented. The construction of a *final ' -typed term* of any type-driven translation is obtained by applying a relevant *translation rule* to each inference rule of a given derivation in its turn.
Now, the translation rules are introduced inductively as follows:

I. *with respect to an AS derivation:*

axioms

(1) $a \rightarrow a$, if a is not s;

are to be translated into:

c_a;

rules of inference

introduction of a functional type in the succedent of a sequent:

(2) $\dfrac{a,T \rightarrow b}{T \rightarrow (a,b)}$ (2′) $\dfrac{T,a \rightarrow b}{T \rightarrow (a,b)}$

both are to be translated into:

$x_a t_b[c_a := x_a]$; where x_a does not occur in the term t_b that corresponds to both: $a,T \rightarrow b$ and $T,a \rightarrow b$.

introduction of a functional type in the antecedent of a sequent:

(3) $\dfrac{T \rightarrow a \quad U,b,V \rightarrow d}{U,T,(a,b),V \rightarrow d}$ and (3′) $\dfrac{T \rightarrow a \quad U,b,V \rightarrow d}{U,(a,b),T,V \rightarrow d}$

both are to be translated into:

$t_d[c_b := c_{(a,b)}(t_a)]$; where $c_{(a,b)}$ does not occur in the terms t_a and t_d which correspond to $T \rightarrow a$ and $U,b,V \rightarrow d$ respectively.

intensionalization of a type in the succedent of a sequent:

(4) $$\frac{T \rightarrow a}{T \rightarrow (s,a)}$$

is to be translated into:

$\lambda x_s t_a$, where t_a corresponds to $T \rightarrow a$.

constraint intensionalization of a type in the antecedent of a sequent:

(5) $$\frac{U,a,V \rightarrow b}{U,(s,a),V \rightarrow b}$$

is to be translated into:

$t_b[c_a := c_{(s,a)}(x_s)]$; where $c_{(s,a)}$ does not occur in the term t_b that corresponds to $U,a,V \rightarrow b$.

Taking into account also the constraint on the AS rule (5) the full impact of the translation rule becomes the following: *in the course of a given type-driven translation the constant c_a is substituted by $x_{(s,a)}(x_s)$; the occurrence of $x_{(s,a)}$ being bound in the corresponding -typed term.*

Comments:

(i) The precise form of the translation rule (3) would be as follows: $t_d[c_b := \lambda x_{(a,b)} x_{(a,b)}(t_a)(c_{(a,b)})]$, by which a slight modification of rule (2) above would also be caused. Clearly, then constants within a final -typed term of a type-driven translation that corresponds to a given AS derivation would occur in precisely the *same order* as antecedent types of the endsequent under consideration. However, we are not primarily interested in this special feature but rather in a comparison of the systems AS and IL_c via their semantics. And in order to make it equivalent but simpler the *normal form* of the above term must be taken for the translation rule (3).

(ii) By the translation rule (4) empty abstractions over x_s can be performed.

(iii) The total of translation rule (5) provides for an auxiliary occurrence of x_s within a bound variable of an intensionally lifted type. In Montague's terminology an extensional operator restricted to the domain of bound variables becomes available by (5).

Next, to better formulate the Ty_2 translation rule which is *obligatorily* applied as the *last step* of any AS type-driven translation if its final λ-typed term is to be *represented* in the *Ty_2 form* the following fact must be proved:

F a c t (4.2.1.):
The final λ-typed term of a type-driven translation that corresponds to an AS derivation with an endsequent Γ, having the form: $a_1, a_2, \ldots, a_n \to b$, is always of the type prescribed by the succedent of Γ and, moreover it is definable by means of distinct constants of each of the types occurring in the antecedent of Γ.
Hence, it can be denoted by: $t_b(c_{a_1}, c_{a_2}, \ldots, c_{a_n})$.

P r o o f:
By induction on the size of an AS derivation and the construction of AS translation rules.

Actually an expanded version of the above fact could easily be justified comprising also a kind of an inverse statement to the one just given:
Suppose $t_b(c_{a_1}, c_{a_2}, \ldots, c_{a_n})$ is a final λ-typed term of a given type-driven translation that corresponds to an AS derivation then the endsequent of the latter is *up to permutation of antecedent types uniquely* determined to be: $a_1, a_2, \ldots, a_n \to b$.

Remark:
Clearly, uniqueness for the above cases respecting also the order of antecedent types and constants in $a_1, a_2, \ldots, a_n \to b$ and $t_b(c_{a_1}, c_{a_2}, \ldots, c_{a_n})$ respectively would result from the precise form for the translation rule (3) as given in comment (i).

To continue with:

Ty_2 translation rule:
For each $i \in \{1,2,\ldots,n\}$, $t_b[c_{a_i} := c_{(s,a_i)}(x_s)]$, where $t_b(c_{a_1},c_{a_2},\ldots,c_{a_n})$ is the final λ-typed term of an AS type-driven translation.

Remark:
Clearly, by the above rule, also referred to as *Gallination*, an *intensional setting* for any AS typed-term is achieved. To put it differently, an implicit reference to indices, characteristic for a one-sorted approach, has thus become explicit, imposed by the relevant terms themselves. Moreover, an additional postulate must be stated still:

Postulate:
The rules (4),(5) as well as Ty_2 translation rule must refer to the *same distinguished variable* x_s in the course of a particular type-driven translation.

Remarks:
Clearly, in $t_b(c_{(s,a_1)}(x_s),\ldots,c_{(s,a_n)}(x_s))$ the distinguished variable x_s may occur free or bound, depending on a given AS derivation to which a type-driven translation is effectively assigned. Furthermore, a *vacuous binding over* x_s may still occur within a given λ-typed term after the Ty_2 translation rule has been applied to it. An example illustrating the latter claim will be given in section (5).

Finally, we present the translation rules

II. *with respect to an IL_c derivation;*
however only the *new* cases, i.e. those involved with a variable x_s, are treated below, with the corresponding IL_c axioms and inference rules being left out as well:

(1) when a is s: x_s

(2) and (2′) when a is s:

$x_s t_b(x_s)$; *t_b being the corresponding λ-typed term of the sequents $s,T \to b$ and $T,s \to b$.*

(3) and (3′) when b is s:

$t_d[x_s := c_{(a,s)}(t_a)]$; *where $c_{(a,s)}$ does not occur in the terms t_a and t_d which correspond to $T \to a$ and $U,b,V \to d$ respectively.*

(5) *identification of the relevant occurrences of s-type variables.*

...

Comments:

No vacuous binding can be produced by the IL_c translation rules. In short, the IL_c fragment can be viewed as the ordinary L-fragment with one liberalization being a consequence of s-type contraction, namely:

by λ-abstraction also more than one occurrence of x_s can be bound simultaneously.

Finally, since the stronger system ILP_{ce} will not be used in applications, later on, the intended translation rules for permutation and t-type expansion will briefly be hinted at, for the sake of generality. By the former one a term, assigned to the premise sequent of the permutation rule, should simply be *preserved*. By the latter one, however, a *free Boolean conjuction* of specific t-type terms is introduced into the term that corresponds to the premise sequent of t-type expansion. More precisely:

$$\frac{U,R,t,V \to a}{U,t,R,t,V \to a}$$

is to be translated into:

$$t_a[c_t := (d_t, e_t)],$$

where d_t and e_t do not occur in the term t_a which corresponds to $U,R,t,V \to a$.

4.3. AS Semantics in Ty_2 Perspective

As was shown above, to any AS derivation a type-driven translation can be assigned. A final λ-typed term of the latter is due to the construction of AS translation rules a well defined Ty_2 term, provided that the Ty_2 translation rule has been performed. In order to determine which fragment of Ty_2 the AS -typed terms after Gallination live in, Gallin's idea of translating *one-sorted* IL terms into *two-sorted* ones is borrowed [5]. Therefore, his Translation Scheme will be presented first:

for each IL term t_b the *translate* of t_b, denoted by $[t_b]'$, is defined as follows:

(i) $$[x^n_a] = x^n_a \;;$$

(ii) $$[c^n_a]' = c^n_{(s,a)}(x_s) \;;$$

(iii) $$[t_{(a,b)}(t_a))]' = [t_{(a,b)}]'([t_a]') \;;$$

(iv) $$[\lambda x_a t_b]' = \lambda x_a[t_b] \;;$$

(v) $$[{}^\wedge t_a]' = \lambda x_s[t_a] \;;$$

(vi) $$[{}^\vee t_{(s,a)}] = [t_{(s,a)}]'(x_s) \;;$$

<u>Remark</u>:

The constants of $[t_a]'$ are the constants $c^n_{(s,b)}(x_s)$ such that c^n_b occurs in t_a. Clearly, the main distinction between the one-sorted and the two-sorted representation of IL terms arises from the fact that a *variable* of the *basic type s* becomes available in the latter case. Thus any application of an intensional and an extensional operator is simply reduced to a λ-abstraction and a functional application over x_s respectively. The essential translation step is actually the substitution of any IL constant by its proper Ty_2 form given inductively on the construction of IL terms as in (ii) to (vi) above.
And that is precisely what the translation rule Ty_2 does within a final λ-typed term of a given type-driven translation.

Further analogies between Gallin´s translation scheme and AS λ-typed terms in their intensional setting are pointed out below.
Since AS rule (4) has the role of an intensional operator, the form of its translation rule has been suggested by (v) above. However, note that $x_s x_a$ can not be a final -typed term of any AS derivation.
Further, AS rule (5) has the role of a restricted extensional operator. The form of its translation rule is thus determined by analogy with (vi) above, suitably restricted to the domain of bound variables.

5. RELATIONS BETWEEN AS and IL_c

The main aim of this section is to clarify a relationship between the systems AS and IL_c making use of their syntactic as well as semantic features.
First of all note, that *neither* of them is a *subsystem* of the other.
Clearly the IL_c axiom $s \rightarrow s$ is not derivable in AS, while the simple AS theorem $e \rightarrow (s,e)$ can not be deduced from IL_c rules. And that is a trivial consequence of the *one-sorted* and the *two-sorted* construction of AS and IL_c respectively.
However, a useful *correspondence* between the two systems can be stated with respect to their *semantic overlap*. The latter being available after the final *Gallination*, i.e. the Ty_2 translation rule, has been performed for each AS λ typed term, obtained by the above translation procedure.
It is easy to see that neither the AS fragment is contained in the IL_c fragment nor vice versa. That fact is justified by the following two counter examples:

(1) AS:

$$\dfrac{\dfrac{\dfrac{a \rightarrow a}{a \rightarrow (s,a)} \qquad b \rightarrow b}{a,((s,a),b) \rightarrow b}}{((s,a),b) \rightarrow (a,b)}$$

Ty_2 rule:

$$\dfrac{\dfrac{\dfrac{\dfrac{c_a}{\lambda x_s c_a} \qquad c_b}{c_{((s,a),b)}(\lambda x_s c_a)}}{\lambda x_a c_{((s,a),b)}(\lambda x_s x_a)}}{\lambda x_a c_{(s,((s,a),b))}(x_s)(\lambda x_s x_a)}$$

since there is an irreducible vacuous binding over x_s within the above λ-typed term the latter can not belong to the IL_c fragment.

(2) IL_c:

$$\dfrac{\dfrac{s \rightarrow s \qquad a \rightarrow a}{(s,a),s \rightarrow a}}{s \rightarrow ((s,a),a)} \qquad\qquad \dfrac{\dfrac{x_s \qquad c_a}{c_{(s,a)}(x_s)}}{\lambda x_{(s,a)} x_{(s,a)}(x_s)}$$

obviously, a *constant-free* λ-typed term can not belong to the AS fragment, since such a term could only be assigned to an endsequent with an empty antecedent, which is not AS derivable.

Note also, that by the IL_c apparatus a *restricted* extensional operator can, in fact, be represented explicitly by the above derived λ-typed term: $\lambda x_{(s,a)} x_{(s,a)}(x_s)$.

In what follows a more sophisticated comparison of both systems will be achieved based on the intersection of the AS fragment after Gallination and the IL_c fragment. Briefly, a *correspondence* between those AS and IL_c *derivations* that can semantically be represented by the same -typed term will be stated. Formally speaking, a *partial* transformation from *equivalence classes* of *AS derivations* to *equivalence classes* of IL_c *derivations* will be introduced; the relevant equivalence relations R_{AS} and R_{IL} defined on AS and IL_c derivations of sequents respectively being induced by the *equality of final -typed terms* assigned to any given AS and IL_c derivations respectively.

The definitions of some of the above discussed notions will be introduced below. To begin with the following

Notation:

Any AS or IL_c derivation d being given (d) denotes a final -typed term in *Ty_2 form* of the type-driven translation corresponding to d.

DEFINITION (5.1):

$d_1 R_{AS} d_2$ if $(d_1) = (d_2)$, where for each i {1,2} d_i is a given AS derivation d_i.

DEFINITION (5.2):

$d_1 R_{IL} d_2$ if $(d_1) = (d_2)$, where for each i {1,2} d_i is a given IL_c derivation.

Notation:

$[d]_{AS}$ and $[d]_{IL}$ denote equivalence classes corresponding to R_{AS}(def. 5.1) and R_{IL}(def. 5.2) respectively.

<u>DEFINITION (5.3)</u>:

T is a partial transformation with
domain: {AS derivations of sequents }/R_{AS} and
codomain: {IL_c derivations of sequents}/R_{IL}.
Suppose d is an AS derivation of some specified sequent.
Then $T([d]_{AS}) = [d']_{IL}$, provided that:

$$\tau(d') = \tau(d);$$

if there is no such IL_c derivation, then T is undefined in $[d]_{AS}$.

<u>Comments</u>:

Clearly T is well-defined. Note also that by the first counter-example above the *partiality* of T is justified while by the second one its *non-surjectivity* is pointed out. Let us proceed with the following important proposition adding a constructive part to the definition of T:

<u>P R O P O S I T I O N (5.4)</u>:

Let d be an AS derivation with the endsequent $a_1, \ldots, a_n \to b$. Then T is defined in $[d]_{AS}$ if and only if precisely one of the following sequents up to permutation of antecedent types:

(1) $(s,a_1), \ldots, (s,a_n) \to b$,

(2) $s, (s,a_1), \ldots, (s,a_n) \to b$,

is the *endsequent* of an IL_c derivation d′, such that:

$$\tau(d') = \tau(d);$$

with (1) applying in case *no* occurrence of x_s in $\tau(d)$ is free and (2) otherwise.

<u>P r o o f</u>:

$(\Rightarrow)$

Suppose T is defined in $[d]_{AS}$. Then by definition (5.3.) there is an IL_c derivation d′ such that $\tau(d') = \tau(d)$.
Now, only the endsequent of d′ remains to be checked, making use of the fact (4.2.1.) together with the final Gallination step $\tau(d)$, and hence $\tau(d')$, can be rewritten as:
$t_b(c_{(s,a_1)}(x_s), \ldots, c_{(s,a_n)}(x_s))$. By the latter term, using

induction on the size of an IL_c derivation and the construction of the IL_c translation rules the succedent of the endsequent is uniquely determined to be b, while its antecedent is determined uniquely up to permutation of the types $(s,a_1),\ldots,(s,a_n)$ and *at most one* basic type s. Clearly, the latter being transmitted to a succedent when all s-type variables occur bound in the final -typed term: $t_b(c_{(s,a_1)}(x_s),\ldots,c_{(s,a_n)}(x_s))$. On the other hand, any free occurrence of s-type variable corresponds to an occurrence of the basic type s in the antecedent of a relevant sequent. In our case, however, contraction is obligatory as a direct consequence of the Postulate imposed on some AS translation rules. Thus, more than one basic s type in the antecedent of an endsequent would be in contradiction with the fact that there is a *single* distinguished variable x_s in $\tau_b(d)$. Hence, depending on the behaviour of x_s in $\tau_b(d)$ precisely one of the sequents stated in (1) and (2) up to permutation of antecedent types is the endsequent of d´. And that was to be proved.

( )

Holds true by definition (5.3.).

Q.E.D.

And finally, using the last proposition some more examples establishing a corresponding IL_c derivation, if it exists, to a given AS derivation are displayed below:

(1) an AS single axiom derivation a a corresponds to the following IL_c derivation:

$$\frac{s,(s,a) \to a}{s \to s \quad a \to a},$$

obviously having the *same* final -typed term in Ty_2 form:

$$c_{(s,a)}(x_s).$$

(2) the AS derivation: $\frac{a \to a}{a \quad (s,a)}$

corresponds to the IL_c derivation given below:

$$\frac{\dfrac{(s,a) \rightarrow (s,a)}{s,(s,a) \rightarrow a}}{s \rightarrow s \quad a \rightarrow a}$$

again with the same final λ-typed term in Ty_2 form:

$$\lambda x_s c_{(s,a)}(x_s).$$

(3) however, the AS derivation:

$$\frac{\dfrac{a \rightarrow a}{a \rightarrow (s,a)}}{a \rightarrow (s,(s,a))}$$

with its final λ-typed term: $\lambda x_s \lambda x_s c_{(s,a)}(x_s)$,
does not have a corresponding IL_c derivation at all. The reason being the vacuous binding over x_s in the final λ-typed term above which can not be produced by IL_c translation rules.
Let us conclude this part by a remark that a suitable *restriction* of the transformation T (def. 5.3.) as well as a restricted version of the last proposition will be used in an application discussed next.

6. APPLICATIONS OF IL_c TO MONTAGUE GRAMMAR

It is well-known that in PTQ Montague defined a recursive correspondence between categories of English and IL-types. Thus, all expressions of a given category are assigned the same semantic type which uniquely determines the corresponding denotational domain. However, the proper semantic description of elements of a given category is essentially dependent on the *degree of intensionality* that a certain expression might convey. Taking as an example the category of transitive verbs one can easily distinguish among basically extensional elements and non-basically extensional ones such as "find" and "seek". While the latter can be interpreted "de re" or "de dicto", only the first interpretation is adequate for the former verb. Thus a *fine structure* concerning the *intended dose of intensionality* of specific expressions is imposed on every category and has to be taken into account in order to obtain an adequate interpretation. Montague solved the problem by the introduction of meaning postulates into his semantic frame. Since their use is somewhat round-about, an alternative dynamic approach will be proposed here based on type derivations and meanings assigned to them. Before going into details the general strategy to be pursued in this section is given.

First of all, Montague´s Meaning Postulates as well as some other relevant aspects of PTQ [12] will be presented with their proposed analysis in the system AS and its semantics. One of the main reasons for the auxiliary system to have been constructed in a one-sorted perspective is just to provide a natural transition to Montague´s intensional system.

Further on, it will be shown how the system IL_c with its semantics can be applied to Montague Grammar achieving the same results as the auxiliary system AS before. The last statement will formally be proved by making use of a suitable restriction of the partial transformation T

(def.5.3.) as well as a restricted version of the proposition (5.4.). Finally, some relevant examples analyzed by the IL_c apparatus will be displayed.
Starting from PTQ let any linguistic category be assigned the type produced by the well-known recursive correspondence:

<u>DEFINITION (6.1)</u>

$$h: Cat \to Types \text{ , such that}$$

$$h(e) = e \text{ , } h(t) = t \text{ , } h(B/A) = h(B//A) = ((s,h(A)),h(B))$$

We continue with an exposition of Montague's

<u>Meaning postulates</u>

MP 1: $\exists u \Box[u = \alpha]$,

where α translates any entity expression;

MP 2: $\Box[\delta(x) \to \exists u[x = {}^{\vee}u]]$,

where δ translates any basically extensional member of B_{CN}.

MP 3: $\exists M \forall x \Box[\delta(x) \leftrightarrow M\{{}^{\vee}x\}]$,

where δ translates any basically extensional member of B_{IV}.

MP 4: $\exists S \forall x \forall P \Box[\delta(x,P) \leftrightarrow P\{{}^{\wedge}\lambda y S\{{}^{\vee}x, {}^{\vee}y\}\}]$,

where δ translates any basically extensional member of B_{TV}.

MP 5: $\forall P \exists M \forall x \Box[\delta(x,P) \leftrightarrow M\{{}^{\vee}x\}]$,

where δ translates any non-basically extensional member of B_{TV}.

MP 6: $\forall p \exists M \forall x \Box[\delta(x,p) \leftrightarrow M\{{}^{\vee}x\}]$,

where δ translates any member of $B_{IV/t}$.

MP 7: $\forall R \exists M \forall x \Box[\delta(x,R) \leftrightarrow M\{{}^{\vee}x\}]$,

where δ translates any member of $B_{IV//IV}$.

MP 8: $\Box\exists G \forall P \forall Q \forall x \Box[\delta(P)(Q)(x) \leftrightarrow P\{{}^{\wedge}\lambda y[[{}^{\vee}G]({}^{\vee}y)(Q)(x)]\}]$,

where δ translates any basically extensional member of $B_{IAV/T}$.

Throughout the above exposition of the meaning postulates the following convention is used:

Variables that occur in the MP's are of the types:

u	e
x,y	(s,e)
M	(s,(e,t))
S	(s,(e,(e,t)))
P	(s,((s,((s,e),t)),t))
p	(s,t)
R	(s,((s,e),t))
G	(s,(e,((s,((s,e),t)),((s,e),t))))
Q	(s,((s,e),t))

The brace convention "{...}" indicates an introduction of "´" immediately in front of the predicate to which the brace convention has been applied.

Remark:

Montague uses his meaning postulates to restrict the universe of all models to just the *admissible ones*, so that the lexical items of certain categories can be interpreted in an adequate way.

Clearly a discrepancy between an intended semantic domain of a lexical item and the one that is recursively assigned to it arises from the definition (6.1), where a recursive correspondence between linguistic categories and types is defined so as to treat adequately the *most intensional* expressions of each category.

As has already been mentioned the approach suggested here will be a dynamic one. Our proposal is the following:

For an arbitrary basic expression x of any category C listed in the above-mentioned postulates there is an AS derivation of the sequent: pt(x) h(C), in such a way that the extension of the corresponding final -typed term is identical with the proper extension of x. That -typed term is to be taken as the carrier of the proper extension of a lexical item x appearing in the MP under consideration. Here "pt(x)" denotes the proper type (determined by the relevant MP, except for lexical items of IAV/T) that *indicates the adequate denotational domain of a given expression x and h is defined by (6.1).*

<u>Remark</u>:
To be precise, the above mentioned λ-typed term is a final term of a type-driven translation that corresponds to a given AS derivation. But for the sake of brevity, the shortened form, as presented above, will be used from now on. Moreover, note that the given proposal is equally valid for obtaining an adequate interpretation of any other expression: not only the ones involved in the above MP's.
Let us add a further comment. Clearly the final λ-typed term corresponding to an AS derivation, in this case to an *intensional lifting* pt(x) $\rightarrow$ h(C), is by fact (4.2.1.) of the *common categorial type h(C)* but is essentially *definable in terms of the basic constant of the type pt(x), called the initial carrier of x*. Thus, the above mentioned λ-typed term becomes the carrier of an adequate interpretation of x. Hence, on the old base of a recursive correspondence between categories and types the *flexibility of interpretation* within each category is achieved by making use of the system AS and its semantic apparatus.
Before implementing the above proposal, note that a particular intensional lifting pt(x) $\rightarrow$ h(C) may have more *semantically distinct* derivations. However, the number of such possible distinct readings still remains to be determined.
Now, two main kinds of *extensionality* of expressions will be treated below, namely those two that the meaning postulates are involved with :
1. *extensionality with respect to argument position in general,*
2. *extensional first-order definability (full or partial).*

The necessity to define the former originates from the uniform correspondence between categories and types constructed in an intensional perspective. Thus an element of category B/A is translated into the corresponding IL term of type ((s,h(A)),h(B)) which is fit to operate on the dual of some term of type h(A). But this should only be the case when the initial expression of category B/A creates an opaque context or else its extension has already an intensional conceptual structure. Thus the remainder of the elements of category B/A must actually be treated as being of type (h(A),h(B)). Montague solved this semantic discrepancy by means of meaning postulates (3),(5),(6) and (7) presented above.

In this case a final -typed term assigned to a specific AS derivation of the sequent:

$$(h(A),h(B)) \cdot ((s,h(A)),h(B))$$

will be taken as the carrier of the proper extension of any expression which should *essentially* have a semantic type (h(A),h(B)). In working out the above case a more familiar notation for types is used , with a and b standing for h(A) and h(B) respectively.

Starting with an AS derivation of the sequent:

$$(a,b) \quad ((s,a),b)$$
$$(a,b),(s,a) \cdot b \quad (2')$$
$$(s,a) \cdot a \qquad\qquad b \quad b \quad (3')$$
$$a \cdot a \quad (5)$$

The corresponding type-driven translation becomes:

$$\cdot x_{(s,a)} c_{(s,(a,b))}(x_s)(x_{(s,a)}(x_s))$$
$$\cdot x_{(s,a)} c_{(a,b)}(x_{(s,a)}(x_s))$$
$$c_{(a,b)}(c_{(s,a)}(x_s))$$
$$c_{(s,a)}(x_s) \qquad\qquad c_b$$
$$c_a$$

<u>Comment</u>:

The above final -typed term is clearly of *common categorial type* ((s,a),b) but is essentially *definable in terms of the basic constant of type* (a,b), being already represented in its Ty_2 form. Thus *extensionality of argument position* is preserved:

$$\lambda x_{(s,a)} c_{(s,(a,b))}(x_s)(x_{(s,a)}(x_s))[{}^\wedge\lambda y_s c_{(s,a)}(y_s)] \Rightarrow_\beta$$
$$\Rightarrow_\beta c_{(s,(a,b))}(x_s)({}^\wedge\lambda y_s c_{(s,a)}(y_s)(x_s)) \Rightarrow_\beta$$
$$\Rightarrow c_{(s,(a,b))}(x_s)(c_{(s,a)}(x_s)).$$

What this shows is that the modally closed term of type (s,a), namely $\lambda y_s c_{(s,a)}(y_s)$, is clearly extensionalized after the second β-conversion has been carried out. Using montagovian terminology: "down-up cancellation" of the argument is performed. Clearly, an analogous reasoning can be used for any intensional constant of type (s,a) occurring as the argument in the last example.
Next, observe, that $\lambda x_{(s,e)} c_{(s,(e,t))}(x_s)(x_{(s,e)}(x_s))$ is just a special case of the above final λ-typed term, namely the carrier of proper extensions of basically extensional elements of the category IV.
Postulates (5),(6) and (7) require the so-called *subject extensionality* referring to non-basically extensional elements of TV as well as to arbitrary expressions of IV/t and IV//IV. Now, the general case which makes it possible to avoid the three postulates mentioned above will be derived at length.
To begin with an AS derivation of the sequent:

$$(A,(e,t)) \to (A,((s,e),t))$$
$$(A,(e,t)),A \to ((s,e),t) \quad (2')$$
$$A \to A \qquad (e,t) \to ((s,e),t)) \quad (3')$$
$$(s,e),(e,t) \to t \quad (2)$$
$$e,(e,t) \to t \quad (5)$$
$$e \to e \qquad t \to t \quad (3)$$

The corresponding type-driven translation is presented below:

$$\lambda x_A \lambda x_{(s,e)} c_{(s,(A,(e,t)))}(x_s)(x_A)(x_{(s,e)}(x_s))$$
$$\lambda x_A \lambda x_{(s,e)} c_{(A,(e,t))}(x_A)(x_{(s,e)}(x_s))$$
$$x_{(s,e)} c_{(A,(e,t))}(c_A)(x_{(s,e)}(x_s))$$
$$c_A \qquad \lambda x_{(s,e)} c_{(e,t)}(x_{(s,e)}(x_s))$$
$$c_{(e,t)}(c_{(s,e)}(x_s))$$
$$c_{(e,t)}(c_e)$$
$$c_e \qquad c_t$$

Obviously the *subject position* in question is *extensionalized*: which was to be achieved.
Thus taking (s,((s,((s,e),t)),t)) , (s,t) and (s,((s,e),t)) for A successively in the above final λ-typed term the corresponding carriers for proper extensions of basic expressions listed in the three postulates are obtained.

The second kind of extensionality arises from so-called *extensional first-order reducibility*. It involves expressions of an intensional higher-order semantic type recursively assigned to a given category but with an intended interpretation that is *fully* or *partially extensional first order*. In particular this holds for extensional elements of the categories TV and IAV/T respectively. Thus the translation of such an expression is definable in terms of:

(i) *a fully extensional first-order* ·*-typed term*

(ii) *a partially extensionalized* ·*-term of a reduced order,*

the former and the latter being the *initial carrier* of an adequate interpretation of an extensional TV and IAV/T expression respectively.
Thus a fully extensional interpretation will be induced to basically extensional TV elements by using our dynamic strategy as follows:
An AS derivation of the sequent determined by MP (4) is given below:

```
   (e,(e,t))  · ((s,((s,((s,e),t)),t)),((s,e),t))
  (s,((s,((s,e),t)),t)),(e,(e,t))  · ((s,e),t)  (2)
   ((s,((s,e),t)),t),(e,(e,t))  · ((s,e),t)   (5)
   ((s,((s,e),t)),t),(e,(e,t)),(s,e)  · t   (2′)
   (e,(e,t)),(s,e)    (s,((s,e),t))       t  · t   (3′)
   (e,(e,t)),(s,e) · · ((s,e),t)   (4)
   (s,e),(e,(e,t)),(s,e)  · t   (2)
     e,(e,(e,t)),(s,e)  · t   (5)
        e,(e,(e,t)),e  · t   (5)
e  · e    (e,t),e  · t   (3)
e  · e      t  · t   (3′)
```

The final -typed term corresponding to the above AS derivation is the following:

$x_a\, y_{(s,e)} x_a(x_s)(\, x_s\, x_{(s,e)} c_{(s,(e,(e,t)))}(x_s)(x_{(s,e)}(x_s))$
$(y_{(s,e)}(x_s)))$,

where the type (s,((s,((s,e),t)),t) is abbreviated with a.

<u>Comment</u>:

As expected, the above -term is of the prescribed higher-order type common to the entire category TV. But on the other hand it is constructed in terms of a fully extensional first-order constant of the proper type (e,(e,t)), again being represented in its Ty_2 form. Moreover, observe that the -typed term of our proposal is analogous to the Montagovian IL term: P xP{ yS(x,y)} derived in accordance with postulate (4) by double -abstraction.

To sum up: a translation of any basically extensional lexical item of category TV is always definable in terms of a basic constant $c_{(s,(e,(e,t)))}(x_s)$ assigned to a given lexical item as the *initial* carrier of its fully extensional intended interpretation.

To better grasp the above claim an exposition of the determination of a particular Ty formula, the carrier of the intended "de re" (referential) reading of the sentence "John seeks a unicorn", will be given next. Note that in our approach the quantification rule (T14) can simply be replaced by the proper functional application rule. Here the translation procedure is displayed:

a $Q\, P\, x[Q(x_s)(x)\ P(x_s)(x)]$

unicorn $y c_{(s,(e,t))}(x_s)(y(x_s))$

a unicorn $P\, x[c_{(s,(e,t))}(x_s)(x(x_s)) \wedge P(x_s)(x)]$

seek $R\, yR(x_s)(\wedge x_s\, z c_{(s,(e,(e,t)))}(x_s)(z(x_s))(y(x_s)))$

seek a unicorn

$y\, x[c_{(s,(e,t))}(x_s)(x(x_s)) \wedge$
$c_{(s,(e,(e,t)))}(x_s)(x(x_s))(y(x_s))]$

John $QQ(x_s)(\, x_s c_{(s,e)}(x_s))$

John seeks a unicorn $x[c_{(s,(e,t))}(x_s)(x(x_s)) \wedge$
$c_{(s,(e,(e,t)))}(x_s)(x(x_s))(c_{(s,e)}(x_s))]$

<u>Remark</u>:

P,Q,R,x,y,z are variables of the types (s,((s,e),t)), (s,((s,e),t)), (s,((s,((s,e),t)),t)), (s,e), (s,e) and (s,e) respectively while $c_{(s,(e,t))}(x_s)$, $c_{(s,(e,(e,t)))}(x_s)$ are the initial carriers of "unicorn" and and the "de re" version of "seek" respectively.

Note that the above formula is logically equivalent to the Montagovian *reduced* translation of the same sentence , namely: u_e[unicorn′(u_e) seek′(u_e)(j)]

Finally, the translation of "John" deserves some special attention. It is well-known that Montague treated proper names semantically in the same way as quantified expressions [12]. Moreover, his first meaning postulate demands the initial carrier of a specific proper name to have a constant intension. Together with MP2, where an *analogous* condition for basically extensional items of B_{CN} is expressed, that seems to be the only claim which have to remain postulated in the present approach as well. However, see section (7) for further directions as regards eventually dismissing the first two postulates. Thus, $c_{(s,e)}(x_s)$ must have an index-invariant extension in the -typed term:

$$QQ(x_s)(\ x_s c_{(s,e)}(x_s)),$$

which is the final -typed term that can be assigned to the following AS derivation of the sequent:

e ((s,((s,e),t)),t)

(s,((s,e),t)),e t (2)

((s,e),t),e t (5)

e (s,e) t ·· t (3′)

e · e (4)

<u>Remark</u>:

Note that the above derived sequent is an instance of the intensional version of the well-known "Montague rule".

To continue with the second kind of extensionality in the category IAV/T. The proposal in the present paper is slightly different from the montagovian one. Instead of the type (e,((s,((s,e),t)),((s,e),t))) prescribed by postulate (8) here its less intensional rival,

$(e,((s,((s,e),t)),(e,t)))$, is chosen to indicate the adequate denotational domain of extensional prepositions. Clearly the only difference results from the additional demand for subject position of the latter to be extensional. And that sounds quite reasonable since the same demand has been proposed for other categories by Montague himself.
To begin with a specific AS derivation of the sequent:

$$
\begin{array}{l}
(e,(A,(e,t))) \to (B,(A,((s,e),t))) \\
B,(e,(A,(e,t))) \to (A,((s,e),t)) \quad (2) \\
B,(e,(A,(e,t))),A \to ((s,e),t) \quad (2') \\
B,(e,(A,(e,t))),A,(s,e) \to t \quad (2') \\
((s,((s,e),t)),t),(e,(A,(e,t))),A,(s,e) \to t \quad (5) \\
(e,(A,(e,t))),A,(s,e) \to (s,((s,e),t)) \qquad t \to t \quad (3') \\
(e,(A,(e,t))),A,(s,e) \to ((s,e),t) \quad (4) \\
(s,e),(e,(A,(e,t))),A,(s,e) \to t \quad (2) \\
e,(e,(A,(e,t))),A,(s,e) \to t \quad (5) \\
e,(e,(A,(e,t))),A,e \to t \quad (5) \\
e \to e \qquad (A,(e,t)),A,e \to t \quad (3) \\
A \to A \qquad (e,t),e \to t \quad (3') \\
e \to e \qquad t \to t \quad (3')
\end{array}
$$

where A and B denote $(s,((s,e),t))$ and $(s,((s,((s,e),t)),t))$ respectively.
The final λ-typed term corresponding to AS derivation above is the following:

$$\lambda x_B \lambda x_A \lambda x_{(s,e)} x_B(x_s)(\lambda x_s \lambda y_{(s,e)} c_{(s,(e,(A,(e,t))))}(x_s)(y_{(s,e)}(x_s))(x_A)(x_{(s,e)}(x_s)).$$

Actually the latter is analogous to the one derived by triple λ-abstraction of the formula :

$$P\{{}^{\wedge}\lambda y[[G]({}^{\wedge}y)(Q)({}^{\wedge}x)]\}.$$

which is up to extensionality of the subject position identical to the one prescribed by the postulate (8).
Thus, by making use of our Lambek-like dynamic mechanism, all but the first two meaning postulates are actually superflous.

In what follows it will be pointed out that the same results concerning Montague MP's can also be achieved by making use of the corresponding IL_c derivations. In order to see that, the transformation T def.(5.3.) *restricted* to the domain:
{AS derivations of sequents of the form a b}/R_{AS},
i.e. with a *single antecedent type* modulo equality of final -typed terms assigned to them, will be used. Next, it will be shown that, in this case, each equivalence class in the domain consists of derivations of *precisely one* such sequent.

F a c t (6.2.):
Each element of the restricted domain of T consists of derivations of *precisely one* AS theorem.

P r o o f:
Suppose d_1 and d_2 are some specific AS derivations of the sequents a b and d e respectively and let $d_1 R_{AS} d_2$. Then by definition (5.1.) $(d_1) = (d_2)$ holds. More specifically, by fact (4.2.1.) the above equality may be rewritten as: $t_b(c_a) = t_e(c_d)$.
Both equalities: a = d and b = e follow immediately. And that was to be proved.

By making use of the proposition (5.4.) it will be shown that the restriction of T, in fact, *is defined* in each previously treated AS derivation, now taken as a representative. Thus the corresponding IL_c derivation, having the same final -typed term, can equally well be embedded into the previously given dynamic proposal.
Let us now start implementing the above strategy on the specific PTQ examples.
An IL_c derivation of the sequent (a,b) ((s,a),b) in the *Ty_2 intensional setting* is given below:

$$(s,(a,b)),s, \rightarrow ((s,a),b)$$
$$(s,(a,b)),s,s \rightarrow ((s,a),b) \quad (contr.)$$
$$s \rightarrow s \qquad (a,b),s \rightarrow ((s,a),b) \quad (3')$$
$$(a,b),s,(s,a) \rightarrow b \quad (2')$$
$$s,(s,a) \rightarrow a \qquad b \rightarrow b \quad (3')$$
$$s \rightarrow s \quad a \rightarrow a \quad (3)$$

with the corresponding type-driven translation:

$$x_{(s,a)}c_{(s,(a,b))}(x_s)(x_{(s,a)}(x_s))$$
$$\lambda x_{(s,a)}c_{(s,(a,b))}(y_s)(x_{(s,a)}(x_s))$$
$$x_{(s,a)}c_{(a,b)}(x_{(s,a)}(x_s))$$
$$c_{(a,b)}(c_{(s,a)}(x_s))$$
$$c_{(s,a)}(x_s) \qquad c_b$$
$$x_s \qquad c_a$$

Thus, the above IL_c derivation has the *same* final λ-typed term as the initially treated AS derivation.
To continue with an IL_c derivation of the sequent $(A,(e,t)) \rightarrow (A,((s,e),t))$ in the Ty_2 intensional setting:

$$s,(s,(A,(e,t))) \rightarrow (A,((s,e),t))$$
$$s,(s,(A,(e,t))),s \rightarrow (A,((s,e),t)) \quad (contr.)$$
$$s \rightarrow s \qquad s,(A,(e,t)) \rightarrow (A,((s,e),t)) \quad (3')$$
$$s,(A,(e,t)),A \rightarrow ((s,e),t) \quad (2')$$
$$A \rightarrow A \qquad s,(e,t) \rightarrow ((s,e),t) \quad (3')$$
$$(s,e),s,(e,t) \rightarrow t \quad (2)$$
$$s \rightarrow s \qquad e,(e,t) \rightarrow t \quad (3')$$
$$e \rightarrow e \qquad t \rightarrow t \quad (3)$$

with the corresponding final λ-typed term:

$$\lambda x_A \lambda x_{(s,e)}c_{(s,(A,(e,t)))}(x_s)(x_A)(x_{(s,e)}(x_s)).$$

Next is an LC_c derivation of the sequent $(e,(e,t)) \rightarrow ((s,((s,((s,e),t)),t)),((s,e),t))$ in the Ty_2 intensional setting:

s,(s,(e,(e,t))) ((s,((s,((s,e),t)),t)),((s,e),t))

(s,((s,((s,e),t)),t)),s,(s,(e,(e,t))), ((s,e),t) (2)

s s ((s,((s,e),t)),t),(s,(e,(e,t))) ((s,e),t) (3′)

((s,((s,e),t)),t),(s,(e,(e,t))),(s,e) t (2′)

(s,(e,(e,t))),(s,e) (s,((s,e),t)) t t (3′)

s,(s,(e,(e,t))),(s,e) ((s,e),t) (2)

s,(s,(e,(e,t))),s,(s,e) ((s,e),t) (contr.)

s s s,(e,(e,t)),(s,e) ((s,e),t) (3′)

(s,e),s,(e,(e,t)),(s,e) t (2)

(s,e),s,(e,(e,t)),s,(s,e) t (contr.)

s s e,(e,(e,t)),s,(s,e) t (3′)

s s e,(e,(e,t)),e t (3)

e e (e,t),e t (3)

e e t t (3′)

The -typed term corresponding to the above IL_c derivation is as follows:

$x_a\ y_{(s,e)} x_a(x_s)(\ x_s\ x_{(s,e)} c_{(s,(e,(e,t)))}(x_s)(x_{(s,e)}(x_s))(y_{(s,e)}(x_s)))$, where the type (s,((s,((s,e),t)),t) is abbreviated with a.

To continue with an IL_c derivation of the sequent e ((s,((s,e),t)),t) in the Ty_2 intensional setting:

s,(s,e) ((s,((s,e),t)),t)

(s,((s,e),t)),s,(s,e) t (2)

s s ((s,e),t),(s,e) t (3′)

(s,e) (s,e) t t (3)

s,(s,e) e (2)

s s e e (3)

with: $x_a x_a(x_s)(\ x_s c_{(s,e)}(x_s))$ where a denotes the type (s,((s,e),t)).

An IL_c derivation of the last sequent (e,(A,(e,t))) (B,(A,((s,e),t))) in the Ty_2 intensional setting is given below:

$$\begin{array}{l}
s,(s,(e,(A,(e,t)))) \to (B,(A,((s,e),t))) \\
B,s,(s,(e,(A,(e,t)))) \to (A,((s,e),t)) \quad (2) \\
B,s,(s,(e,(A,(e,t)))),A, \quad ((s,e),t) \quad (2') \\
B,s,(s,(e,(A,(e,t)))),A,(s,e) \to t \quad (2') \\
s \to s \quad ((s,((s,e),t)),t),(s,(e,(A,(e,t)))),A,(s,e) \to t \; (3') \\
(s,(e,(A,(e,t)))),A,(s,e) \quad (s,((s,e),t)) \quad t \to t \quad (3') \\
(s,(e,(A,(e,t)))),A,(s,e),s \quad ((s,e),t) \quad (2') \\
(s,(e,(A,(e,t)))),s,A,(s,e),s \to ((s,e),t) \quad (contr) \\
s \to s \quad (e,(A,(e,t))),A,(s,e),s \to ((s,e),t) \quad (3') \\
(s,e),(e,(A,(e,t))),A,(s,e),s \to t \quad (2) \\
(s,e),s,(e,(A,(e,t))),A,(s,e),s \to t \quad (contr.) \\
s \to s \quad e,(e,(A,(e,t))),A,(s,e),s \to t \quad (3') \\
s \to s \quad e,(e,(A,(e,t))),A,e \to t \quad (3') \\
e \to e \quad (A,(e,t)),A,e \to t \quad (3) \\
A \to A \quad (e,t),e \to t \quad (3') \\
e \to e \quad t \to t \quad (3')
\end{array}$$

The final λ-typed term assigned to the above IL_c derivation is given below:

$$\lambda x_B \lambda x_A \lambda x_{(s,e)} x_B(x_s)(\lambda x_s \lambda y_{(s,e)} c_{(s,(e,(A,(e,t))))}(x_s)(y_{(s,e)}(x_s))(x_A)(x_{(s,e)}(x_s)).$$

Thus the application of the system IL_c to Montague Grammar has been worked out.

7. F U R T H E R   D I R E C T I O N S

In what follows some further questions and conjectures will be hinted at.

(1) Clearly, ILP_{ce} is a natural extension of an extensional version of Lambek calculus [2] primarily furnishing the latter by a proper *intensional setting*. In order to achieve real *intensional transitions* an axiom scheme a (s,a) with an intended translation rule of the form $x_s x_a$, should be added to the above given calculus and its semantics. Thus also intensional lowering of sequents like ((s,a),b) (a,b) becomes derivable within the system. By making use of a prescribed translation procedure the final -typed term would turn up to be the following: $x_a c_{((s,a),b)}(x_s x_a)$. Obviously, by each of the newly available -typed terms the conditions stated in MP1 and MP2 respectively are now expressible with the aid of Lambek apparatus alone.

(2) Given our discussion of MP's, a general definition of *extensionality* of expressions across categori..s is desirable for possible further linguistic use as well as its intrinsic logical interest.

(3) The point of view in this paper has been that structural rules of a system need not always be defined for all types but only for certain subclasses (e.g. basic tyes). This possible restriction on structural rules provides for finer distinctions in the usual Categorial Hierarchy, as presented by Wansing [13].

(4) The system ILP_c seems similar to the "Intensional Lambek Calculus" of Morrill [1·] which explores a connection with modal logic. It remains to prove or disprove the equivalence of both systems.

(5) As already mentioned in subsection (2.1.), ILP_{ce} seems also applicable to the recent "Dynamic Montague Grammar" proposed in [7].

These questions all point toward a more general research program, namely the Proof Theory and Model Theory of:

Intensional Type Theory

(for some further directions see Gallin [5], van Benthem [3]).

8. R E F E R E N C E S

[1] H. Barendregt, The Lambda Calculus.Its Syntax and Semantics, North-Holland, 1981.

[2] J. van Benthem, The Lambek Calculus, in R. Oehrle et al., eds.,1988.

[3] J. van Benthem, What is Extensionality?, to appear in I. Bodnar et al.,eds., 1989, Annals of the Lorand Eötvös Univ., Budapest.

[4] J. M. Dunn, Relevance Logic and Entailement, in HPL, vol. III., Gabbay & Guenthner (eds.), Reidel, Dordrecht, 1986.

[5] D. Gallin, Intensional and Higher-Order Modal Logic, North-Holland, Amsterdam, 1975.

[6] S. C. Kleene, Introduction to Metamathematics, D. van Nostrand, New York, Toronto, 1952.

[7] J. Groenendijk & M. Stokhof, Dynamic Montague Grammar, ITLI, Prepublication Series, Amsterdam, 1989.

[8] S. Kripke, The Problem of Entailement, Journal of Symbolic Logic 24, 324.

[9] M. Moortgat, Categorial Investigations, Logical and Lingustic Aspects of the Lambek Calculus, Foris, Dordrecht, 1988.

[10] G. Morrill, Intensionality, Boundedness, and Modal Logic, Centre for Cognitive Science, Univ. of Edinburgh, 1989.

[11] H. Schwichtenberg, Some Applications of Cut Elimination, in HML, Jon Barwise (ed), North-Holland, Amsterdam, New York, Oxford, 1977.

[12] R.H.Thomason (ed.), Formal Philosophy, Selected Papers of R. Montague, Yale, New Haven, London, 1975.

[13] H. Wansing, Relevant Quasi-Deductions, Weak Implicational Logics, and Operational Semantics, erscheint in Schriftenreihe der Joachim Jungius Gesellschaft, Hamburg, 1989.