Discontinuity and the Lambek Calculus

Mark Hepple

Department of Computer Science, University of Sheffield, Regents Court, Portobello Street, Sheffield, UK. Email: hepple@dcs.sheffield.ac.uk

Introduction

This paper is concerned with the treatment of discontinuous constituency within Categorial Grammar. In particular, I address the problem of providing an adequate formalisation of categorial connectives proposed by Moortgat (1988), which are useful for handling certain forms of discontinuous constituency. Despite some interesting proposals, a satisfactory logic for these connectives has so far remained elusive. I will provide such a logic, using an approach that falls within the general framework of *labelled deductive systems* (Gabbay, 1991), employing novel methods for reasoning about linear order in resource usage. The approach is illustrated by linguistic applications for extraction, pied-piping and quantification.

The Lambek calculus

Our general framework is the associative Lambek calculus (**L**: Lambek, 1958), a system which falls within the class of formalisms known as Categorial Grammars. The set of types is freely generated from a set of primitive (atomic) types (e.g. {s, np, ...}), using binary infix operators $\backslash$, $/$, $\bullet$. The 'meaning' of these connectives in **L** is fixed by a semantics for the logic, based on a (semigroup) algebra of strings $(\mathcal{L},\cdot)$, i.e. where $\cdot$ is an associative, non-commutative binary operator, with two-sided identity ε, and $\mathcal{L}$ is the set of non-empty ($\neq \varepsilon$) strings over some vocabulary. An interpretation function $[\![]\!]$ assigns some subset of $\mathcal{L}$ to each type, satisfying the conditions below for complex types and type sequences. A type combination $\mathrm{X}_1,...,\mathrm{X}_n \Rightarrow \mathrm{X}_0$ *holds* in a model $((\mathcal{L},\cdot), [\![]\!])$, if $[\![\mathrm{X}_1,...,\mathrm{X}_n]\!] \subseteq [\![\mathrm{X}_0]\!]$, and is *valid* if it is true in all models. There are several formulations of **L** that all realise this same meaning for the connectives.[1]

$$[\![\mathrm{X}\bullet\mathrm{Y}]\!] = \{x\cdot y \in \mathcal{L} \mid x \in [\![\mathrm{X}]\!] \wedge y \in [\![\mathrm{Y}]\!]\}$$

$$[\![\mathrm{X}/\mathrm{Y}]\!] = \{x \in \mathcal{L} \mid \forall y \in [\![\mathrm{Y}]\!].\, x\cdot y \in [\![\mathrm{X}]\!]\}$$

$$[\![\mathrm{Y}\backslash\mathrm{X}]\!] = \{x \in \mathcal{L} \mid \forall y \in [\![\mathrm{Y}]\!].\, y\cdot x \in [\![\mathrm{X}]\!]\}$$

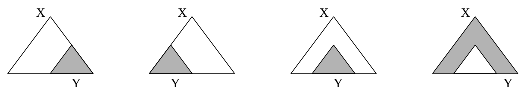
$$[\![\mathrm{X}_1,...,\mathrm{X}_n]\!] = \{x_1\cdot\ldots\cdot x_n \in \mathcal{L} \mid x_1 \in [\![\mathrm{X}_1]\!] \wedge \ldots \wedge x_n \in [\![\mathrm{X}_n]\!]\}$$

[1] The alternative formulations include e.g. sequent (Lambek 1958), proof net (Roorda 1991), and natural deduction systems (Morrill *et al.* 1990, Barry *et al.* 1991). Alternative formulations carry different advantages, e.g. natural deduction is well suited for linguistic presentation, whereas proof nets have benefits for automated theorem proving.

Discontinuous type constructors

The Lambek calculus is a purely concatenative system: where any two types are combined, the string of the result is arrived at by concatenating the strings of the types combined. This point is illustrated graphically in (1a,b), for the Lambek functors, where (following Moortgat, 1991) each triangle represents a result string, and unshaded and shaded areas represent functor and argument strings, respectively.

(1) (a) X/Y (b) Y\X (c) X↑Y (d) X↓Y



Prefixation Suffixation Extraction Infixation

Various linguistic phenomena, however, suggest the existence of discontinous constituency, i.e. situations the result string from combining two constituents is not produced by concatenating the component strings. (See e.g. Bach, 1981.) Moortgat (1988) suggests augmenting **L** with two *discontinuous* type constructors. An *extraction* functor X↑Y is one whose argument corresponds to a non-peripheral (or more precisely, *not-necessarily* peripheral) substring of the result of combination, as in (1c). An infixation functor X↓Y itself corresponds to a non-peripheral substring of the result of combination, as in (1d). Given these intuitive characterisations, two options arise for the meaning of each connective as to whether the point of insertion of one string into the other is free (universal) for fixed (existential). In this paper, I will focus on the existential variants of the connectives, which appear to be the most linguistically useful, and whose interpretive conditions are as follows:

$$[\![\mathrm{X}\!\uparrow\!\mathrm{Y}]\!] = \{x \mid \exists x_1, x_2.\ x = x_1\cdot x_2 \wedge \forall y \in [\![\mathrm{Y}]\!].\ x_1\cdot y\cdot x_2 \in [\![\mathrm{X}]\!]\}$$

$$[\![\mathrm{X}\!\downarrow\!\mathrm{Y}]\!] = \{x \mid \forall y \in [\![\mathrm{Y}]\!].\exists y_1, y_2.\ y = y_1\cdot y_2 \wedge y_1\cdot x\cdot y_2 \in [\![\mathrm{X}]\!]\}$$

Previous proposals

Each connective should have two inference rules: a rule of proof (showing how to derive a type containing the connective), and a rule of use (showing how to employ such a type). This indicates a possible eight inference rules that we might hope to state (i.e. proof/use × universal/existential × infixation/extraction). Various attempts have been made to provide a logic for the discontinuous type constructors, but all have proved unsuccessful or unsatisfactory in some way or another.

Moortgat (1988), for example, uses an ordered sequent calculus framework, which allows only two of the possible eight rules to be stated: a rule of proof for existential $\uparrow$, and a rule of use for universal $\downarrow$. Moortgat (1991) uses a proof method in which types are not ordered in proof representations, where linear order constraints and consequences are instead handled using a system of string labelling, i.e. types are associated with string terms, which are explicitly manipulated by inference rules. This approach allows two further rules to be stated, but the four expressible rules are distributed one per connective, i.e. a complete logic is not given for even any one connective. As Versmissen (1991) notes, Moortgat's string label system does not allow the recording of a specific position for inserting one string into another, as would seem to be required.

Morrill & Solias (1993) avoid this latter problem by augmenting the string labelling algebra with a non-associative pairing operator $\langle .,. \rangle$, allowing labels such as $\langle s_1, s_2 \rangle$, indicating an insertion point in between s_1 and s_2. This system allows versions of $\uparrow$ and $\downarrow$ operators to be specified, but ones whose interpretive definitions differ from Moortgat's. The non-associativity of pairing gives rise to limited flexibility for the system in terms of the type combinations that can be derived, and even the types that can be constructed, e.g. no functor $(X{\uparrow}Y)/Z$, where a $\uparrow$ argument is not the first sought, is allowed.

Labelled deduction & Lambek calculus

I next develop a formulation of **L** which can be extended to allow for the (existential) discontinuity connectives. Our starting point is a lambda term semantics for implicational **L** due to Buszkowski (1987), based on the well known Curry–Howard interpretation of proofs (Howard, 1969).[2] This uses a bidirectional variant of the lambda calculus whose basic terms are directionally typed variables. If t is a term of type $Y\backslash X$ (resp. X/Y), and u one of type Y, then $(ut)^l$ (resp. $(tu)^r$) is a term of type X. If v is a variable of type Y, and t a term of type X, then $\lambda^l v.t$ (resp. $\lambda^r v.t$) is a term of type $Y\backslash X$ (resp. X/Y). A semantics for implicational **L** is given by the class of terms which satisfy the conditions: (B1) each subterm contains a free variable, (B2) no subterm contains > 1 free occurrence of any variable, (B3) each λ^l (resp. λ^r) binds the leftmost (resp. rightmost) free variable in its scope.

This semantics can be used in formulating (implicational) **L** as a *labelled deductive system* (LDS: Gabbay, 1991).[3] Labels are terms of the directional lambda system, and propagation of labels is via application and abstraction in the standard manner. Natural deduction rules labelled in this way are as follows:

(2)
$$\frac{A/B : a \quad B : b}{A : (ab)^r}/E \qquad \frac{\begin{matrix}[B : v] \\ A : a\end{matrix}}{A/B : \lambda^r v.a}/I$$

$$\frac{B : b \quad B\backslash A : a}{A : (ba)^l}\backslash E \qquad \frac{\begin{matrix}[B : v] \\ A : a\end{matrix}}{B\backslash A : \lambda^l v.a}\backslash I$$

We can ensure that only deductions appropriate to (implicational) **L** are made by requiring that the label that results with any inference is a term satisfying Buszkowski's three conditions. To facilitate testing this requirement, I use a function Σ, which maps from label terms to the string of their free variables occurring in the left-right order that follows from type directionality (giving what I call a *marker* term). A notion of 'string equivalence' ($\doteq$) for marker terms is defined by the axioms:

$(\doteq .1)$	$x{\cdot}(y{\cdot}z) \doteq (x{\cdot}y){\cdot}z$
$(\doteq .2)$	$x \doteq \varepsilon{\cdot}x$
$(\doteq .3)$	$x \doteq x{\cdot}\varepsilon$

Σ is recursively specified by the following clauses (where FV returns the set of free variables in a term), *but* it is defined for all and only those terms that satisfy Buszkowski's three conditions.[4] Thus, we can ensure correct deduction by requiring of the label that results with each inference that there exists some marker term m such that $\Sigma(a) = m$.

$(\Sigma.1)$ $\quad \Sigma(v) = v \quad$ where $v \in Vars$

$(\Sigma.2)$ $\quad \Sigma((ab)^l) = \Sigma(a){\cdot}\Sigma(b)$
where $\mathrm{FV}(a) \cap \mathrm{FV}(b) = \emptyset$

$(\Sigma.3)$ $\quad \Sigma((ab)^r) = \Sigma(a){\cdot}\Sigma(b)$
where $\mathrm{FV}(a) \cap \mathrm{FV}(b) = \emptyset$

$(\Sigma.4)$ $\quad \Sigma(\lambda^l v.a) = \beta$
where $\mathrm{FV}(\lambda^l v.a) \neq \emptyset,\ \ \Sigma(a) \doteq v{\cdot}\beta$

$(\Sigma.5)$ $\quad \Sigma(\lambda^r v.a) = \beta$
where $\mathrm{FV}(\lambda^r v.a) \neq \emptyset,\ \ \Sigma(a) \doteq \beta{\cdot}v$

The following proofs illustrate this LDS (using $t \overset{\Sigma}{\mapsto} m$ as shorthand for $\Sigma(t) \doteq m$, to indicate a significant marker

[2]Under the Curry–Howard interpretation (Howard, 1969), logical formulas are regarded as types of expressions in typed lambda calculus, with atomic formulas corresponding to basic types, and a formula $A{\rightarrow}B$ to the type of functions from A to B. It is demonstrable that the set of formulas for which there exists some correspondingly typed lambda term is precisely the theorems of the implicational fragment of intuitionistic logic. Thus, typed lambda calculus provides a *semantics* for implicational intuitionistic logic, i.e. an independent characterisation of 'valid deduction', just as the algebraic semantics of **L** provides an independent characterisation of validity for that system. Semantics for various other logics can be given in terms of *classes* of typed lambda terms, i.e. subsets of the typed lambda terms which satisfy certain stated criteria. van Benthem (1983) provides a lambda semantics for the system **LP**, a commutative variant of **L**. Wansing (1990) provides lambda semantics for a range of sublogics of intuitionistic logic. The Curry–Howard interpretation so permeates categorial work that the terms "formula" and "type" have become almost interchangeable. Note that I have slightly modified Buszkowski's notation.

[3]In labelled deduction, each formula is associated with a label, which records information of the use of resources (i.e. assumptions) in proving that formula. Inference rules indicate how labels are propagated, and may have side conditions which refer to labels, using the information recorded to ensure correct inferencing. Evidently, the Moortgat (1991) and Morrill & Solias (1993) formalisms are LDSs.

[4]Condition B2 is enforced by the requirement on the application cases of Σ. Conditions B1 and B3 are enforced by the first and second requirement on the abstraction cases of Σ, respectively.

equivalence):

$$\dfrac{\dfrac{\mathrm{X/Y}:x \qquad \dfrac{\mathrm{Y/Z}:y \quad [\mathrm{Z}:z]}{\mathrm{Y}:(yz)^r}\,/\mathrm{E}}{\mathrm{X}:(x(yz)^r)^r}\,/\mathrm{E}}{\mathrm{X/Z}:\lambda^r z.(x(yz)^r)^r}\,/\mathrm{I} \qquad \overset{\Sigma}{\mapsto}\ x\cdot y\cdot z$$

$$\dfrac{\dfrac{\mathrm{X/(Y/(Z\backslash Y))}:x \qquad \dfrac{\dfrac{[\mathrm{Z}:z] \quad [\mathrm{Z\backslash Y}:y]}{\mathrm{Y}:(zy)^l}\,\backslash\mathrm{E}}{\mathrm{Y/(Z\backslash Y)}:\lambda^r y.(zy)^l}\,/\mathrm{I}}{\mathrm{X}:(x\ \lambda^r y.(zy)^l)^r}\,/\mathrm{E}}{\mathrm{X/Z}:\lambda^r z.(x\ \lambda^r y.(zy)^l)^r}\,/\mathrm{I} \qquad \overset{\Sigma}{\mapsto}\ z\cdot y, \quad \overset{\Sigma}{\mapsto}\ x\cdot z$$

This system can be extended to cover product using the inference rules (3), and the additional Σ clauses shown following (with the obvious implicit extensions of the directional lambda system, and of Buszkowski's semantics). Labelling of [$\bullet$I] inferences is via pairing, and that of [$\bullet$E] inferences uses an operator adapted from Benton *et al.* (1992), where a term $[b/v\bullet w].a$ implicitly represents the substitution of b for $v{+}w$ in a. This rule is used in (4).

(3)
$$\dfrac{\begin{matrix}[\mathrm{B}:v]\ [\mathrm{C}:w] \\ \mathrm{A}:a\end{matrix} \qquad \mathrm{B{\bullet}C}:b}{\mathrm{A}:[b/v\bullet w].a}\,\bullet\mathrm{E} \qquad\qquad \dfrac{\mathrm{A}:a \quad \mathrm{B}:b}{\mathrm{A{\bullet}B}:\langle a,b\rangle}\,\bullet\mathrm{I}$$

$$(\Sigma.6) \qquad \Sigma(\langle a,b\rangle) = \Sigma(a)\cdot\Sigma(b) \quad \text{where } \mathrm{FV}(a)\cap\mathrm{FV}(b)=\emptyset$$

$$(\Sigma.7) \qquad \Sigma([b/v\bullet w].a) = \beta_1\cdot\Sigma(b)\cdot\beta_2 \quad \text{where } \mathrm{FV}(a)\cap\mathrm{FV}(b)=\emptyset, \ \Sigma(a) \doteq \beta_1\cdot v\cdot w\cdot\beta_2$$

(4)
$$\dfrac{\dfrac{\dfrac{\dfrac{\mathrm{X/Y/Z}:x \quad [\mathrm{Z}:z]}{\mathrm{X/Y}:(xz)^r}\,/\mathrm{E} \qquad [\mathrm{Y}:y]}{\mathrm{X}:((xz)^r y)^r}\,\backslash\mathrm{E} \qquad \mathrm{Z{\bullet}Y}:w}{\mathrm{X}:[w/z\bullet y].((xz)^r y)^r}\,\bullet\mathrm{E}}{\mathrm{X/(Z{\bullet}Y)}:\lambda^r w.([w/z\bullet y].((xz)^r y)^r)}\,/\mathrm{I} \qquad \overset{\Sigma}{\mapsto}\ x\cdot z\cdot y, \quad \overset{\Sigma}{\mapsto}\ x\cdot w$$

Labelled deduction & discontinuity

This approach can be extended to allow for existential $\uparrow$ and $\downarrow$. These connectives have standard implicational inference rules, using additional distinguished operators for labelling (with superscript e for extraction and i for infixation):

(5)
$$\dfrac{\mathrm{A{\uparrow}B}:a \quad \mathrm{B}:b}{\mathrm{A}:(ab)^e}\,\uparrow\mathrm{E} \qquad \dfrac{\begin{matrix}[\mathrm{B}:v] \\ \mathrm{A}:a\end{matrix}}{\mathrm{A{\uparrow}B}:\lambda^e v.a}\,\uparrow\mathrm{I}$$

$$\dfrac{\mathrm{A{\downarrow}B}:a \quad \mathrm{B}:b}{\mathrm{A}:(ab)^i}\,\downarrow\mathrm{E} \qquad \dfrac{\begin{matrix}[\mathrm{B}:v] \\ \mathrm{A}:a\end{matrix}}{\mathrm{A{\downarrow}B}:\lambda^i v.a}\,\downarrow\mathrm{I}$$

Consider firstly how Σ must be extended for the abstraction cases of the new introduction rules. For a [$\uparrow$I] term such as $\lambda^e v.a$, the relevant Σ case allows v to appear non-peripherally in the marker term of a. For a [$\downarrow$I] term such as $\lambda^i v.a$, v is allowed to be discontinuous in the marker of a (we shall see shortly how this is possible), but requires its components to appear peripherally.

$$(\Sigma.8) \qquad \Sigma(\lambda^e v.a) = \beta_1\cdot\beta_2 \quad \text{where } \mathrm{FV}(\lambda^e v.a)\neq\emptyset, \ \ \Sigma(a)\doteq\beta_1\cdot v\cdot\beta_2$$

$$(\Sigma.9) \qquad \Sigma(\lambda^i v.a) = \gamma \quad \text{where } \mathrm{FV}(\lambda^i v.a)\neq\emptyset, \ \ \Sigma(a)\doteq\beta_1\cdot\gamma\cdot\beta_2, \ \ \beta_1\cdot\beta_2 = v$$

To allow for the new application operators, the marker system must be extended. Recall that the linear order information implicit in labels is projected onto the left-right dimension in markers. With $\uparrow$ and $\downarrow$, however, the possibility exists that either functor or argument is discontinuous in the result of their combination. For strings $x \in [\![\mathrm{X{\uparrow}Y}]\!]$ and $y \in [\![\mathrm{Y}]\!]$, for example, we know there is some way of wrapping x around y to give a result in X, but we do not in general know how the division of x should be made. This problem of *uncertainty* is handled by using operators L and R, where L(m) and R(m) represent *indefinite* but *complementary* left and right subcomponents of the marker term m. (L and R are *not* projection functions.) This idea of the significance of L and R is given content via the additional axiom ($\doteq$.4), which allows that if the complementary left and right subcomponents of a marker appear in appropriate left-right juxtaposition, then the marker's resources may be treated as continuous.[5]

$$(\doteq.4) \qquad \mathrm{L}(x)\cdot\mathrm{R}(x)\doteq x$$

The remaining clauses for Σ then are:

$$(\Sigma.10) \qquad \Sigma((ab)^e) = \mathrm{L}(\Sigma(a))\cdot\Sigma(b)\cdot\mathrm{R}(\Sigma(a)) \quad \text{where } \mathrm{FV}(a)\cap\mathrm{FV}(b)=\emptyset$$

$$(\Sigma.11) \qquad \Sigma((ab)^i) = \mathrm{L}(\Sigma(b))\cdot\Sigma(a)\cdot\mathrm{R}(\Sigma(b)) \quad \text{where } \mathrm{FV}(a)\cap\mathrm{FV}(b)=\emptyset$$

Some example derivations follow:

$$\dfrac{\dfrac{\mathrm{X/Y}:x \quad [\mathrm{Y}:y]}{\mathrm{X}:(xy)^r}\,/\mathrm{E}}{\mathrm{X{\uparrow}Y}:\lambda^e y.(xy)^r}\,\uparrow\mathrm{I} \ \ \overset{\Sigma}{\mapsto}\ x\cdot y\cdot\varepsilon \qquad \dfrac{\dfrac{\mathrm{X/Y}:x \quad [\mathrm{Y}:y]}{\mathrm{X}:(xy)^r}\,/\mathrm{E}}{\mathrm{X{\downarrow}Y}:\lambda^i y.(xy)^r}\,\downarrow\mathrm{I} \ \ \overset{\Sigma}{\mapsto}\ \varepsilon\cdot x\cdot y$$

$$\dfrac{\dfrac{[\mathrm{X{\uparrow}Y}:\mathrm{x}] \quad \mathrm{Y}:\mathrm{y}}{\mathrm{X}:(xy)^e}\,\uparrow\mathrm{E}}{\mathrm{X{\downarrow}(X{\uparrow}Y)}:\lambda^i x.(xy)^e}\,\downarrow\mathrm{I} \qquad \dfrac{\dfrac{[\mathrm{X{\downarrow}Y}:\mathrm{x}] \quad \mathrm{Y}:\mathrm{y}}{\mathrm{X}:(xy)^i}\,\downarrow\mathrm{E}}{\mathrm{X{\uparrow}(X{\downarrow}Y)}:\lambda^e x.(xy)^i}\,\uparrow\mathrm{I}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\mathrm{(X/Y){\uparrow}Z}:x \quad [\mathrm{Z}:z]}{\mathrm{X/Y}:(xz)^e}\,\uparrow\mathrm{E} \qquad [\mathrm{Y}:y]}{\mathrm{X}:((xz)^e y)^r}\,/\mathrm{E}}{\mathrm{X{\uparrow}Z}:\lambda^e z.((xz)^e y)^r}\,\uparrow\mathrm{I}}{\mathrm{(X{\uparrow}Z)/Y}:\lambda^r y\lambda^e z.((xz)^e y)^r}\,/\mathrm{I} \qquad \begin{matrix}\overset{\Sigma}{\mapsto}\ \mathrm{L}(x)\cdot z\cdot\mathrm{R}(x)\cdot y \\ \overset{\Sigma}{\mapsto}\ \mathrm{L}(x)\cdot\mathrm{R}(x)\cdot y \ \doteq\ x\cdot y\end{matrix}$$

[5]This axiom may be seen as stating the limit of what can be said concerning 'uncertainly divided' resources, i.e. only where the uncertainty is eliminated by juxtaposition can the L,R operators be removed, making some otherwise 'hidden' resource visible. Further reasonable axioms (not in practice required here) are $\mathrm{L}(\varepsilon)\doteq\varepsilon$ and $\mathrm{R}(\varepsilon)\doteq\varepsilon$, i.e. the only possible left and right subcomponents of an 'empty' marker are likewise empty.

$$\dfrac{\dfrac{\dfrac{\mathrm{X/Y}:x \qquad \dfrac{[\mathrm{Y{\uparrow}Z}:y] \qquad [\mathrm{Z}:z]}{\mathrm{Y}:(yz)^e}{\uparrow}\mathrm{E}}{\mathrm{X}:(x(yz)^e)^r}/\mathrm{E}}{\mathrm{X{\uparrow}Z}:\lambda^e z.(x(yz)^e)^r}{\uparrow}\mathrm{I}}{(\mathrm{X{\uparrow}Z})/(\mathrm{Y{\uparrow}Z}):\lambda^r y\lambda^e z.(x(yz)^e)^r}/\mathrm{I} \qquad \begin{array}{l} \overset{\Sigma}{\mapsto}\ x{\cdot}\mathrm{L}(y){\cdot}z{\cdot}\mathrm{R}(y) \\ \overset{\Sigma}{\mapsto}\ x{\cdot}y \end{array}$$

Word order and NL semantics

Labels encode both the functional structure and linear order information of proofs, and hence are used in identifying both the NL semantic and word order consequences of combinations. Label terms, however, encode distinctions not needed for NL semantics, but can easily be simplified to terms involving only a single abstractor (λ) and with application notated by simple left-right juxtaposition, e.g.

$$\lambda^r z\lambda^l x.(x(yz)^r)^l \quad \rightsquigarrow \quad \lambda z\lambda x.((yz)x).$$

To determine the linear order consequences of a proof with label a, we might seek a marker m consisting only of concatenated variables, where $\Sigma(a) \doteq m$. These variables would be the labels of the proof's undischarged assumptions, and their order in m would provide an order for the types combined under the proof. Alternatively, for linguistic derivations, we might substitute lexical string atoms in place of variables, and seeker a marker consisting only of concatenated string atoms, i.e. a word string. This method is adequate for basic **L**, but problems potentially arise in relation to the discontinuity connectives.

Consider the transformation $\mathrm{X/Y} \Rightarrow \mathrm{X{\uparrow}Y}$. The connective of the result type does not record all the linear order import of the input type's connective, and neither consequently will the application label operator for a subsequent [↑E]. However, β-normalisation yields a simpler label term whose operators record the linear order information originally encoded in the connectives of the types combined. For example, the following proof includes a subderivation of $\mathrm{X/Y} \Rightarrow \mathrm{X{\uparrow}Y}$. The overall proof term does not simply order the proof's assumptions under Σ (giving marker $\mathrm{L}(x){\cdot}y{\cdot}\mathrm{R}(x)$), but its β-normal form $(xy)^r$ does (giving $x{\cdot}y$).

$$\dfrac{\dfrac{\dfrac{\mathrm{X/Y}:x \qquad [\mathrm{Y}:v]}{\mathrm{X}:(xv)^r}/\mathrm{E}}{\mathrm{X{\uparrow}Y}:\lambda^e v.(xv)^r}{\uparrow}\mathrm{I} \qquad \mathrm{Y}:y}{\mathrm{X}:((\lambda^e v.(xv)^r)\ y)^e}{\uparrow}\mathrm{E}$$

Of course, normalisation can only bring out ordering information that *is* implicit in the types combined. For example, the combination $\mathrm{X{\uparrow}Y}:x, \mathrm{Y}:y \Rightarrow \mathrm{X}:(xy)^e$ is a theorem, but the label $(xy)^e$ does not simply order x and y. However, if we require that lexical subcategorisation is stated only using the standard Lambek connectives, then adequate ordering information will always be encoded in labels to allow simple ordering for linguistic derivations. Alternatively, we could allow discontinuity connectives to be used in stating lexical subcategorisation, and further allow that lexical types be associated with *complex string terms*, constructed using label operators, which encode the requisite ordering information. For example, a word w with lexical type X↑Y might have a string term $\lambda^e v.(wv)^r$, which does encode the relative ordering of w and its argument. A more radical idea is that deduction be made over lexical types together with their (possibly complex) lexical string terms, and that the testing of side conditions on inferences be done on the β-normal form of the end label, so that the implicit ordering information of the lexical string term is 'brought out', extending proof possibilities. Then, the lexical units of the approach are in effect partial proofs or derivations.[6] Such a change would greatly extend the power of the approach. (We shall meet a linguistic usage for this extension shortly.)

Linguistic applications

We shall next briefly consider some linguistic uses of the discontinuity connectives in the new approach. The most obvious role for ↑ is in handling extraction (hence its name). Adapting a standard approach, a relative pronoun might have type rel/(s↑np), i.e. giving a relative clause (rel) if combined with s↑np (a 'sentence missing a NP somewhere'). Note that standard **L** allows only types rel/(s/np) and rel/(np\s), which are appropriate for extraction from, respectively, right and left peripheral positions only. For example, *whom Mary considers _ foolish* can be derived under the following proof. The atom string (6a) results via substitution of lexical string terms in the proof label, and Σ. Substitution of lexical semantics and deletion of directional distinctions gives (6b).

$$\begin{array}{c} \text{(whom)} \qquad \text{(mary)} \qquad \text{(considers)} \qquad\qquad \text{(foolish)} \\ \dfrac{\mathrm{rel}/(\mathrm{s{\uparrow}np}):w \qquad \dfrac{\dfrac{\mathrm{np}:x \qquad \dfrac{\dfrac{((\mathrm{np\backslash s})/\mathrm{adj})/\mathrm{np}:y \quad [\mathrm{np}:u]}{(\mathrm{np\backslash s})/\mathrm{adj}:(yu)^r}/\mathrm{E} \qquad \mathrm{adj}:z}{\mathrm{np\backslash s}:((yu)^r z)^r}/\mathrm{E}}{\mathrm{s}:(x((yu)^r z)^r)^l}\backslash\mathrm{E}}{\mathrm{s{\uparrow}np}:\lambda^e u.(x((yu)^r z)^r)^l}{\uparrow}\mathrm{I}}{\mathrm{rel}:(w\ \lambda^e u.(x((yu)^r z)^r)^l)^r}/\mathrm{E} \end{array}$$

(6) a. *whom·mary·considers·foolish*
b. whom′ (λu.considers′ u foolish′ mary′)

Moortgat (1991) suggests that a (for example) sententially scoped NP quantifier could be typed s↓(s↑np), *if* infixation and extraction could be linked so that infixation was to the position of the 'missing np' of s↑np.[7] Such linkage does not follow from the definitions of the connectives but can be implemented in the present approach by assigning a complex lexical string term, as in the lexical entry (<TYPE,STRING,SEM>):

[6] This idea invites comparisons to formalisms such as *lexicalised tree adjoining grammar* (see Joshi *et al*, 1991), where the basic lexical and derivational units are partial phrase structure trees associated with lexical items.

[7] In the approach of Morrill & Solias (1993) such linkage follows automatically given the interpretive definitions of their connectives. Moorgat (1990,1991) proposes special purpose quantification type constructors.

$<\mathrm{s}{\downarrow}(\mathrm{s}{\uparrow}\mathrm{np}),\ \lambda^e u.(u\ \mathit{someone})^e,\ \mathrm{someone}'>$

Such a string term would result under a 'type raising' transformation such as: np ⇒ s↓(s↑np). The example *John gave someone money* can be derived as follows, with string and semantic results in (7).

$$\dfrac{\overset{\text{(someone)}}{\mathbf{s{\downarrow}(s{\uparrow}np)}:q} \qquad \dfrac{\dfrac{\overset{\text{(john)}}{\mathbf{np}:x} \qquad \dfrac{\dfrac{\overset{\text{(gave)}}{\mathbf{((np\backslash s)/np)/np}:y} \quad [\mathbf{np}:v]}{\mathbf{(np\backslash s)/np}:(yv)^r}/\mathrm{E} \qquad \overset{\text{(money)}}{\mathbf{np}:z}}{\mathbf{np\backslash s}:((yv)^r z)^r}/\mathrm{E}}{\mathbf{s}:(x((yv)^r z)^r)^l}\backslash\mathrm{E}}{\mathbf{s{\uparrow}np}:\lambda^e v.(x((yv)^r z)^r)^l}{\uparrow}\mathrm{I}}{\mathbf{s}:(q\ \lambda^e v.(x((yv)^r z)^r)^l)^r}{\downarrow}\mathrm{E}$$

(7) a. *john·gave·someone·money*
b. someone′ (λv.gave′ v money′ john′)

There is a sense in which this view of quantifiers seems very natural. Quantifiers behave distributionally like simple NPs, but semantically are of a higher type. Raising the string component under the transformation np ⇒ s↓(s↑np) resolves this incompatibility without imposing additional word order constraints.

This account as stated does not allow for multiple quantification,[8] but would if lexical string terms were treated as partial proofs used in assembling larger derivations, as suggested in the previous section.

In interesting test case, combining both movement and scope issues, arises with pied piping constructions, where a *wh*-item moving to clause initial position is accompanied by (or 'pied pipes') some larger phrase that contains it, as in e.g. the relative clause *to whom John spoke*, where the PP *to whom* is fronted. Following Morrill & Solias (1993), and ultimately Morrill (1992), a treatment of pied piping can be given using ↑ and ↓. Again, linkage of infixation and extraction is achieved via complex lexical string assignment. A PP pied-piping relative pronoun might be (rel/(s↑pp))↓(pp↑np) allowing it to infix to a NP position within a PP, giving a functor rel/(s↑pp), i.e. which prefixes to a 'sentence missing PP' to give a relative clause. Hence, for example, *to whom* would have type rel/(s↑pp), and so *to whom John spoke* is a relative clause. The lexical semantics of *whom* will ensure that the resulting meaning is equivalent to the non-pied piping variant *whom John spoke to*.

[8]For example, we might seek to extend the proof just given by abstracting over z in a [↑I] inference, as a basis for adding in a further quantifier, but the current proof label would not license such an inference, due to the presence of the $()^l$ application.

References

Bach, E. 1981. 'Discontinuous Constituents in Generalized Categorial Grammars.' *NELS*, **11**, pp1–12.

Barry, G., Hepple, M., Leslie, N. and Morrill, G. 1991. 'Proof figures and structural operators for categorial grammar'. In *Proc. of EACL–5*, Berlin.

van Benthem, J. 1983. 'The semantics of variety in Categorial Grammar.' Report 83-29, Dept. of Mathematics, Simon Fraser University. Also in W. Buszkowski, W. Marciszewski and J. van Benthem (Eds), *Categorial Grammar*, Vol. 25, Linguistic and Literary Studies in Eastern Europe, John Benjamins. 1988.

Benton, N., Bierman, G., de Paiva, V. & Hyland, M. 1992, 'Term assignment for intuitionistic linear logic.' Technical Report, Cambridge University Computer Laboratory.

Buszkowski, W. 1987. 'The Logic of Types.' In J. Srzednicki (Ed), *Initiatives in Logic*, Martinus Nijhoff Publishers, Dordrecht.

Gabbay, D. 1991. *Labelled deductive systems.* Draft 1991. (to appear: Oxford University Press).

Hepple, M. 1990. *The Grammar and Processing of Order and Dependency: A Categorial Approach.* Ph.D. dissertation, Centre for Cognitive Science, University of Edinburgh.

Howard, W.A. 1969. 'The formulae-as-types notion of construction.' In J.R. Hindley & J.P. Seldin (Eds), *To H.B. Curry, Essays on Combinatory Logic, Lambda Calculus and Formalism*, AP, 1980.

Joshi, A.K., Vijay-Shanker, K. & Weir, D. 1991. 'The convergence of mildly context-sensitive formalisms'. In P. Sells, S. Shieber & T. Wasow (Eds.) *Foundational issues in Natural Language Processing.* MIT Press, Cambridge MA.

Lambek, J. 1958. 'The mathematics of sentence structure.' *American Mathematical Monthly*, **65**.

Moortgat, M. 1988. *Categorial Investigations: Logical and Linguistic Aspects of the Lambek Calculus*, Foris, Dordrecht.

Moortgat, M. 1990. 'The quantification calculus.' In Hendriks, H. and Moortgat, M. (Eds), *Theory of Flexible Interpretation.* Esprit DYANA Deliverable R1.2.A, Institute for Language, Logic and Information, University of Amsterdam.

Moortgat, G. 1991. 'Generalized Quantification and Discontinuous type constructors'. To appear: W. Sijtsma & A. van Horck (Eds) *Proc. Tilburg Symposium on Discontinuous Constituency.* De Gruyter.

Morrill, G. 1992. 'Categorial Formalisation of Relativisation: Pied Piping, Islands and Extraction Sites.' Research Report, Dept. de Llenguatges i Sistemes Informàtics, Universitat Politècnica de Catalunya.

Morrill, G. & Solias, M.T. 1993. 'Tuples, Discontinuity, and Gapping in Categorial Grammar.' In: *Proc. of EACL–6*, Utrecht.

Roorda, D. 1991. *Resource Logics: Proof Theoretical Investigations.* Ph.D. Dissertation, Amsterdam.

Versmissen, K. 1991. 'Discontinuous type constructors in Categorial Grammar.' ms. OTS, Universiteit Utrecht, Netherlands.

Wansing, W. 1990. 'Formulas-as-types for a hierarchy of sublogics of Intuitionistic Propositional Logic.' ms. Institut für Philosophie, Freie Universität Berlin.