Combinatory Categorial Grammar
and
Linguistic Diversity

ESSLLI 2005, August 8–12, Edinburgh

Cem Bozsahin* & Mark McConville†

*Middle East Technical University, Ankara
†The University of Edinburgh
Outline

CCG as a theory:
- Substantive aspects (CB)
- Formal aspects (MM)
- Limits on NL variation (MM)
- Constructions (CB)

Linguistic Diversity:
- Ergative and accusative languages (CB)
  (and OpenCCG implementations)
- Morphology and organisation of the lexicon (MM)
Substantive aspects:

Syntactic types, semantic types

and principles

Cem Bozsahin
Categorial Grammar (CG) is unique in its treatment of the notion of “possible category”, and how syntax can be made completely type-dependent, rather than structure-dependent, with transparent semantics.

Combinatory Categorial Grammar (CCG) is unique among categorial grammars in its treatment of constituency, (un)bounded dependencies, and binding.

There have been no syntactic variables in CCG ever since its inception (Ades and Steedman, 1982, written 1979), and no other grammar than a lexicalised grammar, therefore there can be no locus for movement-like operations to arise.

With the Minimalist Program’s (MP) recent use of one binary combinator (Merge), and attempts to subsume Move under Merge (Epstein et al., 1998), theories seem to converge, differences narrowing down to the notion of “possible category.” (see Steedman 2005a for more discussion).
.. 'units' and 'grammatical facts' are only different names for different aspects of the same general fact: the operation of linguistic oppositions. So much so that it would be perfectly possible to tackle the problem of units by beginning with grammatical facts. (F. de Saussure, *Cours de Linguistique Générale*, 1916:168).

Four criteria for tests for grammatical constituents and constituent boundaries: (Noam Chomsky, *LSLT*, 1956/1979:210)

a. The rule for conjunction

b. Intrusion of parenthetical expressions

c. Ability to enter transformations

d. Certain intonational features
- **Constituency** is something to be accounted for and explained,

- rather than defined by a theory.

- Only (c) above is a theory-specific definition of constituency; others are empirical criteria.

**to-infinitive**  *n.* A conventional label for an infinitival verb phrase preceded by the formative *to*, as in *Lisa wants to buy a BMW*. In traditional grammar, such a sequence as *to buy* was regarded as a single form, the so-called ‘infinitive’ of the verb *buy*, but this analysis is rejected by all contemporary theories of grammar: all possible tests point to the conclusion that the sequence *buy a BMW* is a constituent (a verb phrase), and hence to the conclusion that *to buy* is not a constituent of any kind. (R.L. Trask, Dictionary of Grammatical Terms in Linguistics, 1993).
Mary wants to **read** and **discuss** the stories of Edgar Allan Poe

Mary wants **to read** and **to discuss** the stories of Edgar Allan Poe

Mary wants **to try to read** and **to be able to discuss** the stories of Edgar Allan Poe

Mary wants—I think—to read the stories of Edgar Allan Poe

Mary wants to discuss—or read? can’t remember—the stories of Edgar Allan Poe

Mary wants—I think—to **discuss**—or **read**? can’t remember—the stories of Edgar Allan Poe
Clearly we may say that if presentations, expressible thoughts of any sort whatever, are to have their faithful reflections in the sphere of meaning-intentions, then there must be a semantic form which corresponds to each presentational form. [...] And if the verbal resources of language are to be a faithful mirror of all meanings possible \textit{a priori}, then language must have grammatical forms at its disposal which give distinct expression, i.e. sensibly distinct symbolisation, to all distinguishable meaning-forms. (\textbf{Edmund Husserl}, \textit{Logical Investigations}, 1890)

The lexicon of a given language is a finite subset of the set of all categories, subject to quite narrow restrictions that ultimately stem from limitations on the variety of semantic types with which the syntactic categories are paired in the lexicon. \textbf{Mark Steedman}, \textit{The Syntactic Process}, 2000:32
• **Category**: a label (symbol, feature bundle etc.) for the purpose of capturing a linguistic distinction.

• If they were to be lexicalised, there should be countably infinitely many syntactic categories.

• Every syntactic category determines a set of structured meanings (logical form—LF), because it is made of substantive categories (nouns, verbs etc.).

• Every LF (and some language specific settings, such as directionality) determine a set of syntactic categories.
Systematicity is not only vertical (cross-categorial), but horizontal as well: categories and semantic types correspond in non arbitrary ways.
The lexicon is really an appendix of the grammar, a list of basic irregularities. This is all the more evident if meanings are taken into consideration, since the meaning of each morpheme belongs to it by an arbitrary tradition. (L. Bloomfield, *Language*, 1933:274)

- Modern incarnation of grammar-lexicon dichotomy appears to originate from Bloomfield—and adopted by Chomsky (1995:130)

  “We distinguish the lexicon from the computational system of a language, the syntax in a broad sense (including phonology).”

- For Chomsky, any systematicity in the lexicon is a missed generalisation, i.e. it belongs to grammar, cf. (1965:167) through (1995:130).
Even when lexicon is considered to be a part of the computational system, the dichotomy prevails (Reinhart and Siloni, 2004, p.160):

“Unlike approaches that decrease the role of the lexicon from an operative component to a list of items (for example, Borer, in this volume; Embick, in this volume; Marantz 1997), we assume that the lexicon is a computational component, where derivational operations can apply.”

“Further, we attribute the somewhat different nature of reflexive verbs in Hebrew, Dutch, and English vs. Romance to the distinct component of grammar in which the operation applies: lexicon vs. syntax.”
CCG argues that this dichotomy gets in the way of our understanding of how syntax (in narrow—combinatory—sense) can shape possible human lexicons.

i.e. The substantive categories of the lexicon are instantiations of the formal categories of Universal Grammar (UG), therefore represent a continuum.

Languages differ only in their lexicons (called Radical Lexicalism by Karttunen 1989).*

By implication, any combinatory difference must be lexically specifiable.

*We might call grammar-lexicon ‘grammaticon’ to avoid the orthographic dichotomy as well.
Lexicalising a Surface Grammar:
A Game of Algebra

Phrase Structure Grammars and corresponding machinery as models of surface syntax encode constituency (and sometimes, order), but some information will be redundantly specified:
\[
\begin{align*}
S & \rightarrow \text{NP VP} \\
\text{VP} & \rightarrow V_{iv} \\
\text{VP} & \rightarrow V_{tv} \text{ NP} \\
V_{iv} & \rightarrow \text{slept} \quad e \mapsto t \\
V_{tv} & \rightarrow \text{read} \quad e \mapsto (e \mapsto t)
\end{align*}
\]

*read*'s subcategorisation for an object \(\text{NP}\) is defined twice: in its lexical category \((V_{tv})\) and in the \(\text{VP}\) rule.

NB. Its (normalised) semantic type is non-redundantly specified.

\(\text{VP}\) is the functor, which, applied to a leftward \(\text{NP}\), yields an \(S\); i.e. \(\text{VP} = (S \backslash \text{NP})\)

\(V_{iv} = \text{VP} = (S \backslash \text{NP})\)

\(V_{tv} = \text{VP/NP} = (S \backslash \text{NP})/\text{NP}\)
Therefore, the only ineliminable parts of the grammar above are *slept* and *read*:

\[
\text{slept} \overset{\text{def}}{=} \text{V}_{iv} = (S\backslash\text{NP})
\]

\[
\text{read} \overset{\text{def}}{=} \text{V}_{tv} = (S\backslash\text{NP})/\text{NP}
\]

Assuming \( S \) to be of semantic type \( t \), and \( \text{NP} \) to be \( e \), a fully interpretable equivalent of the grammar above is

\[
\begin{align*}
\text{slept} & \ := \ S\backslash\text{NP} \quad e \mapsto t \\
\text{read} & \ := \ (S\backslash\text{NP})/\text{NP} \quad e \mapsto (e \mapsto t)
\end{align*}
\]
Predicate-Argument Structure:
Syntacticised argument structure (LF)

This is a point of departure for CCG.

Dowty, Jacobson, Szabolcsi follow the strict Montagovian tradition of not predicing anything on LF (LF is dispensible).

Steedman, Baldridge, Bozsahin, White follow Montague in the sense that syntax is purely type-dependent and entirely blind to both derivation and LF (as opposed to structure-dependent on either derivation or LF), but LF is involved in formulating eg. binding and NL generation.

Type-logical Grammar maintains Lambek calculus for base logic (i.e. no combinators in ‘base’) and a strict Montagovian regime, but pure type-dependence is the unifying theme (see Morrill 1994 and Oehrle 2000 for TLG).
The semantic type \( e \mapsto t \) corresponds to a predicate over a single argument, e.g. \( \text{sleep'} x \)

The \( x \) argument is associated with the syntactic type \( \text{NP} \), via the lambda binding

\[
slept := S\text{NP}: \lambda x.\text{sleep'} x
\]

Binding asymmetries of these syntacticised arguments are encoded in the predicate-argument structure.
\[ \text{read} := (S\setminus \text{NP})/\text{NP}: \lambda x \lambda y. \text{read}' xy \]

\[ \begin{array}{c}
\text{read}' \\
\lambda y \\
\text{read}' x
\end{array} \]

\[ \text{pred}' x_n x_{n-1} \cdots x_1 \] reflects the primacy of \( n \) arguments where \( x_{i-1} \) immediately dominates \( x_i \), yielding the following hierarchy for the linearised notation:

\[ \begin{array}{c}
\text{pred}' \\
\lambda x_{n-1} \\
\cdots
\end{array} \\
\text{pred}' x_n \\
\begin{array}{c}
\lambda x_n \\
\lambda 1
\end{array} \]
(The terms 1, 2, 3 are borrowed from Relational Grammar (Blake, 1990), but there can be no promotion or demotion of them in CCG’s monostratal architecture.)
The semantics of read, \( \lambda x.\, \lambda y.\, \text{read'}\, xy \), is the same as the lexical entry in the PSG.

Actually, lambda bindings are a matter of convenience to associate syntax and semantics; they can all be eliminated.

Eta-conversion:

\[ \lambda x. Fx \leftrightarrow^\eta F \quad \text{(if } x \text{ does not occur free in } F \text{)} \]

e.g. \( \lambda x.1 + x \eta = 1+ \)

\[ \lambda x_1\lambda x_2\lambda x_3.\, \text{show'}\, x_2x_1x_3 = \lambda x_1\lambda x_2\lambda x_3.\, \text{C (show'}\, \text{BCI})x_3x_2x_1 \]
B, C and I are **combinators** (Curry and Feys, 1958), whose operation can be defined in lambda-calculus, e.g. $I \overset{def}{=} \lambda x.x$

<table>
<thead>
<tr>
<th>Commutator</th>
<th>Composer</th>
<th>Identity</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C f a b = f b a$</td>
<td>$B f g a = f(ga)$</td>
<td>$I a = a$</td>
</tr>
</tbody>
</table>

$$\lambda x_1 \lambda x_2 \lambda x_3. C \left( show' BCI \right) x_3 x_2 x_1 \overset{\eta}{=} \lambda x_2 \lambda x_3. C \left( show' BCI \right) x_3 x_2$$

$$\overset{\eta}{=} \lambda x_3. C \left( show' BCI \right) x_3 \overset{\eta}{=} C \left( show' BCI \right)$$
\( show' x_2 x_1 x_3 \overset{?}{=} C (show' BCI) x_3 x_2 x_1 \)

\[
C (show' BCI) x_3 x_2 x_1 = \]

\( show' BCI x_2 x_3 x_1 = \)

\( show' BC (I_2) x_3 x_1 = \)

\( show' C (I_2) x_3 x_1 = \)

\( show' (I_2) x_1 x_3 = \)

\( show' x_2 x_1 x_3 = \)

\( CI = T \quad (aka. \ C_*) \)

\( CI a f = T a f = f a \)
Don’t try this at home; this is just to show that CCG is a **combinatory** theory of syntax-semantics because combinators do show up in LF; not because derivations are decorated with the names of the combinators.

Categorial Grammar is a theory of grammar in which the form-meaning relation is conceived as a transparent correspondence between the surface-syntactic and semantic combinatorics (Jacobson, 1996).

The other place combinators show up is the universal grammar of syntactic projection (more on this later).

Universal Grammar of CCG can be conceived as a specialisation of combinators for language; combinators might be at work at other symbol-driven cognitive activity, eg. planned sequence of actions (see Steedman 2002 for more).
Why \((S\backslash NP)/NP_{acc}/NP_{dat}\): \(\lambda x_1 \lambda x_2 \lambda x_3. show'/ x_2x_1x_3\) for *show*?

*John showed Mary herself*  

*John showed herself Mary*

*John showed Mary to herself*  

*John showed herself to Mary*

The order in which syntactic arguments are taken is different than their obliqueness order.

This is another point of departure in CCG. Bach (1979) and Dowty (1996) allow “wrap” in syntax, and Szabolcsi (1989) simulates wrap so that obliqueness is preserved in the syntactic types (therefore, no use of LF).

However, the following example (from Szabolcsi) is problematic without extra categories for reflexives, and binding in VSO languages remains unaccounted for:
John introduced [[Mary to himself] and [Susan to herself]]

Wrap in syntax: \((S/\text{NP})/\text{NP} \_w \text{PP}: introduce\)'

\(\text{cf. } (S/\text{NP})/\text{PP} / \text{NP}\)

‘Introduced Mary’ must compose before the reflexive (allowed by wrap) so that subject or object can bind the reflexive without the use of LF.

Argument cluster coordination does not follow as a theorem of coordination working on same types (more on this later).

The version we assume is LF (and only lexical) use of “wrap” to capture obliqueness order in LF.

\((S/\text{NP})/\text{NP}_{\text{acc}}/\text{NP}_{\text{dat}} : \lambda x_1 \lambda x_2 \lambda x_3 . show' x_2 x_1 x_3\)

Binding restrictions of VSO languages now hold at LF, as in all others including OVS and OSV languages, with one category for reflexives (for English).
Besides theoretical implications, wrapped categories no longer represent surface order of constituents.
Type determinism significantly constrains the possible syntax-semantics correspondence in the lexicon (and predicts some LFs depending on the availability of some categories).

The principle of Categorial Type Transparency:  
(Steedman 2000:36)

For a given language, the semantic type of the interpretation together with a number of language-specific directional parameter settings uniquely determines the syntactic category of a category.

This principle works both ways: The semantic type of an interpretation is entirely determined by the syntactic type.
e.g. $S$ is $t$ $\quad \text{NP is } e$ $\quad N$ is $e \mapsto t$ (property)

If $A$ is of type $\alpha$ and $B$ $\beta$, then $A\backslash B$ and $A/B$ have the semantic type $\beta \mapsto \alpha$

$\text{see} := (S\backslash\text{NP})/\text{NP}: \lambda x \lambda y. (\text{see}' x)y$ \quad $e \mapsto (e \mapsto t)$

The LF $\lambda x \lambda y. (\text{see}' y)x$ for English is ruled out by the following observation: $\textit{John saw himself}$ (subject must LF-command object)

It is the right LF for a VSO language such as Irish (and for the same reason): $(S/\text{NP})/\text{NP}: \lambda x \lambda y. (\text{see}' y)x$

$\lambda x. \text{see}' xx$ is universally disallowed: $*\textit{Heself saw John}$ (only pro-terms—eg. $x$ in $(\text{ana}' x)$ or $(\text{pro}' x)$—can be LF-commanded by themselves, and the first occurrence of $x$ is not a pro-term).
Predicates such as *help* avail some LFs that can materialise different ways under PCTT:

\[ \text{John helped him to fix the car} \]

\[
\frac{((S\backslash NP)/(S_{\text{inf}}\backslash NP))/NP}{e \mapsto ((e \mapsto t) \mapsto (e \mapsto t))}
\]

\[
\lambda y \lambda P \lambda x . \text{help}^l P y x
\]

“Exceptional” case marking verbs pair naturally with *help*-like verbs:

Both \( \lambda y \lambda P \lambda x . \text{pred}^l P y x \) and \( \lambda y \lambda P \lambda x . \text{pred}^l (P y)x \) are possible by PCTT:
John expects him to fix the car

\[(S_{NP}/(S_{inf}\ \NP))/\NP\]

\[e \mapsto ((e \mapsto t) \mapsto (e \mapsto t))\]

\[\lambda y \lambda P \lambda x. expect'(Py)x\]

*Him* \((y)\) gets its case from *expect* as the /\NP-argument of the verb.

It is not a semantic argument of *expect*, but *fix the car* \((P)\).
Can we eliminate all rules? Syncategorematic rules appear to be problem

\[ X \rightarrow X \text{ and } X \]

where \( X \) is any category in the grammar \((N,V,S,NP,VP,PP...)\)

\[ \text{and } = (X/X) \setminus X \]

In a lexicalised grammar, “any category” includes all the lexical categories,

thus \( S, N, V = \{(S/NP), (S/NP)/NP, (S/NP)/NP/NP\}, VP, NP, PP \) coordination is predicted.

Also predicted is the conjunction of infinitely many categories that are derived from the lexical categories (i.e. the closure of the lexicon with respect to all combinatory possibilities)
The simplest combinatory possibility is **function application**:

\[
\begin{align*}
    &X/Y : f \quad Y : a \quad \Rightarrow \quad X : fa \\
    &Y : a \quad X\setminus Y : f \quad \Rightarrow \quad X : fa
\end{align*}
\]

(>)

(<)

which allows lexical and derived VPs to coordinate:

Mary read the book and studied

\[
\begin{array}{c}
(S\setminus NP)/NP \quad NP/N \quad N \\
\downarrow \quad NP \\
\downarrow \quad S\setminus NP
\end{array}
\]
There is no record of history of derivations in CCG.

Therefore, no way to ‘peek’ inside a constituent to extract some information.

Example: a step-by-step derivation
Mary read the book and studied.
Mary read the book and studied.
Mary read the book and studied.

\[ \text{NP \ (S\backslash NP)/NP \ (X\backslash X)/X \ S\backslash NP} \]

\[ \text{NP} \]
Mary read the book and studied.
Mary read the book and studied.
Mary read the book and studied

[S\NP]

\[
\begin{align*}
\text{NP} & \quad \text{read} \quad \text{the} \quad \text{book} \\
\text{(X\ X)/X} & \quad \text{and} \\
\text{S\ NP} & \quad \text{studied}
\end{align*}
\]

\[
\begin{align*}
\text{S\ NP} & \\
\text{(S\ NP)/(S\ NP)} & >
\end{align*}
\]
Mary read the book and studied

(S\NP)(S\NP)
Mary read the book and studied
Mary read the book and studied
Mary read the book and studied

NP

S\NP

S

S
Mary read the book and studied
Mary read the book and studied.

(S\NP)/(S\NP) / (S\NP)/NP / NP/N / N / (X\X)/X / S\NP

(S\NP)/(S\NP) / (S\NP)/(S\NP) / S\NP / S\NP

S

43
Same example in OpenCCG:

cem-english> tccg
Loading grammar from URL: file:/home/bozsahin/openccg/grammars/cem-english/grammar.xml

Enter strings to parse.
Type ':r' to realize selected reading of previous parse.
Type ':h' for help on display options and ':q' to quit.

tccg> Mary read the book and studied
1 parse found.

Parse: s

--------------

(lex)  Mary :- np
(>T)   Mary :- s/@i(s}@inp)
(lex)  read :- s\.np/\np
(lex)  the :- np/^n
(lex)  book :- n
(>)    the book :- np
(>)    read the book :- s\.np
(lex)  and :- s$1\*(s$1)/*(s$1)
(lex)  studied :- s\.np
(>)    and studied :- s\.np\*(s\.np)
(<>)   read the book and studied :- s\.np
(>)    Mary read the book and studied :- s
Strings like the following need more than application, unless we allow phonologically empty elements in the lexicon:

Mary wants to read and to discuss

\[(S_{\text{inf}} \backslash \text{NP})/(S \backslash \text{NP}) (S \backslash \text{NP})/\text{NP}\]

\[\Rightarrow\]

\[(S_{\text{inf}} \backslash \text{NP})/\text{NP}\]

\[\Rightarrow\]

\[(S_{\text{inf}} \backslash \text{NP})/(S \backslash \text{NP}) (S \backslash \text{NP})/\text{NP}\]

\[\Rightarrow\]

\[(S_{\text{inf}} \backslash \text{NP})/\text{NP}\]

\[?\]

\[?\]

the stories of Edgar Allan Poe

Function composition

\[X/Y : f \quad Y/Z : g \Rightarrow X/Z : \lambda x.f(gx) \quad (> B)\]

\[Y/Z : g \quad X/Y : f \Rightarrow X/Z : \lambda x.f(gx) \quad (< B)\]
Type Raising: turning arguments into functions looking for functors looking for such arguments.

\[ \begin{align*}
A : a & \Rightarrow T/(T\setminus A) : \lambda f. f a \\
A : a & \Rightarrow T\setminus(T/A) : \lambda f. f a
\end{align*} \]  

(> T)

where \( A \) is an argument category in the lexicon, and \( T \) is the result in any function category over \( A \) that the grammar licenses.

Combinatory UG only uses the formal categories such as \( X \) and \( Y \),

Lexical rules use substantive categories such as lexical generalisations \( T \) and \( A \) above, which are generalisations over \( S, NP, PP \) etc.
Type raising and composition engenders so-called “non-constituent coordination”.

It is constituent coordination in CCG because the conjuncts are fully interpretable surface constituents:

\[ \text{[Johnson admires]} \quad \text{and} \quad \text{[Monboddo says he detests]} \quad \text{a saxophonist} \]

\[
\frac{\text{S/NP} \rightarrow^B \text{(S/$\star$S)/$\star$S}}{\text{S/NP} \rightarrow^B \text{S/(S/NP)}} \rightarrow \frac{\text{(S/NP)/$\star$(S/NP)}}{\text{(S/NP)}} < \frac{\text{S}}{<}
\]
\[
\begin{array}{ccc}
\text{Johnson} & \text{admires} \\
\text{NP} & (\text{S}/\text{NP})/\text{NP} \\
: \text{johnson}' & : \lambda z \lambda w. \text{admires}' zw \\
\Rightarrow^T & \Rightarrow^T \\
\text{T}/(\text{T}/\text{NP}) & : \lambda P.P \text{johnson}' \\
\Rightarrow^B & \\
\text{S}/\text{NP} & : \lambda x. \text{admires}' x \text{johnson}'
\end{array}
\]

\[
Bfg = \lambda x. f(gx) = \lambda x. \text{admires}' x \text{johnson}'
\]

\[
(gx) = (\lambda z \lambda w. \text{admires}' zw)x = \lambda w. \text{admires}' xw
\]

\[
f(gx) = \lambda P.P \text{johnson}' (\lambda w. \text{admires}' xw) = (\lambda w. \text{admires}' xw) \text{johnson}'
\]

= \text{admires}' x \text{johnson}'
Unbounded compositions of type $S/\text{NP}$ are predicted to be conjoinable with *Johnson admires*

Monboddo says he detests

\[
\begin{array}{c}
\text{NP} \quad (S\backslash \text{NP})/S \\
\text{T}/(T\backslash \text{NP}) \\
\text{S}/S \\
\hline
\text{S}/(S\backslash \text{NP}) \\
\hline
S/\text{NP} \\
\end{array}
\]

\[
\begin{array}{c}
\text{NP} \quad (S\backslash \text{NP})/\text{NP} \\
\text{T}/(T\backslash \text{NP}) \\
\text{S}/S \\
\hline
\text{B} \\
\end{array}
\]
The star modalities (\* and /\*) on the slash allow lexical control of the construction, e.g. like-categories for coordination (Baldridge, 2002):

*player that shoots and he misses

\[(N\backslash N)/(S\backslash NP)\quad S\backslash NP\quad (S\backslash S)/\star S\quad \star S\quad >\]

\[\star S\quad >\quad \star S\quad >\quad B\]

\[\star S\quad >\quad S\backslash NP\]
The slash modalities

- The $\star$ modality is the most restricted and allows only the most basic applicative rules;

- $\diamond$ permits order-preserving associativity in derivations;

- $\times$ allows limited permutation;

- and $\cdot$ is the most permissive, allowing all rules to apply.
Principles constraining universal combinatory syntax:

*The Principle of Adjacency (PA)*:
Combinatory rules may only apply to finitely many phonologically realised and string-adjacent entities.

*The Principle of Consistency (PC)*:
All syntactic combinatory rules must be consistent with the directionality of the principal functor.

*The Principle of Inheritance (PI)*:
If the category that results from the application of a combinatory rule is a function category, then the slash type of a given argument in that category will be the same as the one(s) of the corresponding argument(s) in the input function(s).
Adjacency: syntactic projection via UG rules has combinatorial basis.

Consistency: certain rules cannot be part of UG eventhough they satisfy adjacency:

\[
\begin{align*}
X /_\star Y & \quad \Rightarrow \quad X & (>) \\
Y & \quad \Rightarrow \quad X & (<)
\end{align*}
\]

\[
\begin{align*}
Y & \quad \Rightarrow \quad X & \text{(disallowed)} \\
X /_\star Y & \quad \Rightarrow \quad X & \text{(disallowed)}
\end{align*}
\]
PI eliminates composition rules below, even though PA and/or PC hold:

\[
\begin{align*}
X \triangleleft Y \quad Y \triangleright Z & \Rightarrow X \triangleright Z \\
& \text{ (disallowed)} \\
X \triangleleft Y \quad Y \triangleright Z & \Rightarrow X \otimes Z \\
& \text{ (disallowed)} \\
X \triangleright Y \quad Y \triangleright Z & \Rightarrow X \triangleright Z \\
& (\triangleright B)^* \\
Y \triangleright Z \quad X \triangleright Y & \Rightarrow X \triangleright Z \\
& (\triangleleft B)
\end{align*}
\]

*These are rule schemata showing the compatibility of modalised slashes, rather than rules doing the actual work. The (\triangleright B) rules can be fleshed out as
\[
\begin{align*}
X \triangleleft Y \quad Y \triangleright Z & \Rightarrow X \triangleright Z, \\
X \triangleright Y \quad Y \triangleright Z & \Rightarrow X \triangleright Z, \\
X \triangleright Y \quad Y \triangleright Z & \Rightarrow X \triangleright Z
\end{align*}
\]
etc.
Unlike Lambek (1958) calculus,* the following are possible rules of universal syntactic projection; they satisfy all principles

The crossing functional composition rules

\[ \frac{X \times Y}{Y \times Z} \Rightarrow \frac{X \times Z}{(\text{>}\text{B}_\times)} \]

\[ \frac{Y \times Z}{X \times Y} \Rightarrow \frac{X \times Z}{(\text{<}\text{B}_\times)} \]

*See Moortgat (1997) for a general introduction.
They have a re-ordering effect, but conserve directionality under syntactic projection because of the Principle of Inheritance:
Adjuncts and second arguments can invert order, as e.g. in Heavy NP shift:

\[
\begin{align*}
\text{I introduced to Marcel some very heavy friends} \\
\text{S/(S/NP): } & ((S/NP)/\text{PP}_{\text{TO}})/\text{NP}: S/(S/\text{PP}_{\text{TO}}): S/(S/NP): \\
\lambda p. p \text{ me}' & \lambda x \lambda y \lambda z. \text{introduce}'yxz \lambda q. q \text{ marcel}' & \lambda r. r \text{ friends}' \\
(S/\text{PP}_{\text{TO}})/\text{NP}: & \lambda x \lambda y. \text{introduce}'yx \text{ me}' < \text{B}_x \\
\text{S/NP}: & \lambda x. \text{introduce}'\text{marcel}'x \text{ me}' < \\
\text{S: } & \text{introduce}'\text{marcel}'\text{friends}'\text{me}' <
\end{align*}
\]

Obliqueness of arguments is preserved (i.e. there is one LF), thus we would expect a single category assignment for such lexical items.

Such derivations preserve the binding condition C at the level of logical form as required by \textit{I introduced to each other some very heavy friends.}
Having eliminated rules, we would expect constructions to follow from the lexical categories (of heads and specifiers of syntactic constructions) alone.

Combinatory syntax simply projects lexical properties, including directionality and LF.

**The Principle of Lexical Head Government** (PLHG):
Both bounded and unbounded syntactic dependencies are specified by the lexical syntactic type of their head.

Syntactic derivation is purely syntactic type driven; LF cannot undo a derivation (like GB/MP, and unlike HPSG and LFG).

This is not to say that LF plays no part in shaping the *lexical* syntactic type; cf. PCTT.
Unlike GPSG, HPSG and TAG, CCG also attempts to adhere to lexical economy:

**The Maxim of Head Categorial Uniqueness (HCU):**
A single nondisjunctive lexical category for the head of a given construction specifies both the bounded dependencies that arise when its complements are in canonical position and the unbounded dependencies that arise when those complements are displaced under relativisation, coordination, and the like.
• Johnson *admires Monboddo*.

• the man *that I believe that Johnson admires*

• I believe that Johnson *admires* and you believe that he *despises, the celebrated judge Lord Monboddo*.

In both TAG and GPSG these dependencies are mediated by different initial trees or categories, and in HPSG they are mediated by a disjunctive category.