On the Logic of LGB Type Structures. Part III: Minimalism

MARCUS KRACHT

ABSTRACT. In this paper we shall in detail at Minimalism in the sense of [3]. We shall show how the structures can be formalised in modal logic.

1. Introduction

This paper continues the series on the logic of LGB-type structures. The idea is to provide tools that allow to decide the modal theory of logics arising within generative grammar. The principal reason why I use the term ”LGB-type” is that in contrast to earlier work in generative grammar, LGB does away with deletion. It is important to realise that from LGB onwards the structures that are being generated are rich enough that existence and well–formedness of a derivation can be checked easily on the basis of the final output (LF).

2. Minimalism à la Stabler

We shall first discuss the codification of minimalism in [3]. We shall simplify the mechanism somewhat in order not to generate confusion. The exposition will also introduce our notation which will differ insignificantly form Stabler’s. We shall return to the full program later.

2.1. **Lexicon.** There is an alphabet A of letters, and alphabet $C := \{\odot, \oplus, \ominus, \circledast\}$ of **control symbols** as well as an alphabet B of **basic features**. Furthermore, for every $b \in B$

(1) $$CB = \{\odot\, b, \oplus\, b, \ominus\, b, \circledast\, b : b \in B\}$$

This is the set of **control features**. For homogeneity, we use $\circledast\, b$ rather than just b. Also the change from + to $\oplus$ and - to $\ominus$ is merely typographical (it makes reading the formulae easier). Notice that the features `=X`, `+X` and `-X` are absent since we do not deal with LF-movement in this paper, to make matters not overly complicated. A lexical entry is a member of

(2) $$\mathrm{Lex} := A^*(CB)^*$$

Not all of these sequences make sense, but the mechanism will see to it that they cannot be used in a complete derivation. A **lexicon** L is a subset of Lex. We assume that a lexicon is finite. Finally, there are two binary operators, > and <, from which terms are being formed, so that Struct is the set of binary terms that can be formed over Lex. However, not all of them are well-formed.

Given a lexicon, Term(L) is the smallest set such that

① $L \subseteq \mathrm{Term}(L)$.
② If $s, t \in \mathrm{Term}(L)$ then also $<(s,t) \in \mathrm{Term}(L)$ and $>(s,t) \in \mathrm{Term}(L)$.

Term := Term(Lex). Terms are identified here with trees. The identification is slightly different from Stabler's own. In Stabler's sense, the leaves of a tree have labels in Lex, and complex trees are formed by bracketing two trees into a single one. Intermediate leaves will be labelled only with < or >. It will emerger that the operations merge and move defined below will create not exactly trees over L but trees over

(3) $$P(L) := \{\vec{x}^\frown\vec{y} : \vec{x} \in A^*, y \in (CB)^*, \text{ and there is } \vec{z}.\vec{x}^\frown\vec{y}^\frown\vec{z} \in L\}$$

Thus, $P(L)$ conists of all the prefixes of members of L that contain then entire phonological matrix of that member.

Our trees look a little different. The leaves of the nodes have labels $a \in A$, $\circledast, \odot, \ominus, \oplus$ as well as $b \in B$. All these labels are mutually incompatible. A sequence is coded as a left-branching (ie ascending tree). Notice however that the modal structures will have more relations in them, so that it is perhaps not correct to speak of trees. Details will follow below.

2.2. **Merge and Move.** There are two functions, merge and move. The operation merge is defined as follows. First, we have to define the notion of a *head*. Consider $t \in$ Term.

(1) If $t \in$ Lex then $\mathrm{hd}(t) := t$.
(2) If $t = >(s, s')$ then $\mathrm{hd}(t) := \mathrm{hd}(s')$.
(3) If $t = <(s, s')$ then $\mathrm{hd}(t) := \mathrm{hd}(s)$.

Finally, ft(t) is the last member of hd(t), provided that it contains a member of CB. It is undefined otherwise. Now, let us define the following operation.

(1) If $t \in$ Lex and $t = \vec{x}^\frown \mathrm{ft}(t)$ then $\mathrm{cut}(t) := \vec{x}$.
(2) If $t = >(s,t)$ then $\mathrm{cut}(s,t) = >(s, \mathrm{cut}(t))$.
(3) If $t = <(s,t)$ then $\mathrm{cut}(s,t) = <(\mathrm{cut}(s), t)$.

Now, merge(s, t) is defined in two cases.

① $\mathrm{ft}(s) = {=}b$ and $\mathrm{ft}(t) = \circledast\, b$. Then $\mathrm{merge}(s,t) := <(\mathrm{cut}(s), \mathrm{cut}(t))$.
② $\mathrm{ft}(t) = \odot\, b$ and $\mathrm{ft}(s) = \circledast\, b$. Then $\mathrm{merge}(s,t) := >(\mathrm{cut}(s), \mathrm{cut}(t))$.

Notice that if defined, the result is unique.

Next we look at move. It is a unary operation. It is defined only when $\mathrm{ft}(t) = \oplus\, b$ for some $b \in B$. In that case, we define two operations, ext and int, defined in the following way. $\mathrm{int}(t, b)$ is defined if there is exactly one maximal subterm t such that $\mathrm{ft}(t) = \ominus\, b$. Then $\mathrm{int}(t, b)$ is that maximal subtree. If no such tree exists, $\mathrm{int}(t, b)$ is undefined. Furthermore, $\mathrm{ext}(t, b)$ is the result of removing the occurrence of $\mathrm{int}(t, b)$ from t (that is, replacing it by the empty featureless string $\varepsilon \in \mathrm{Lex}$). Now,

$$\mathrm{move}(t) := {>}(\mathrm{cut}(\mathrm{int}(t, \mathrm{ft}(t))), \mathrm{cut}(\mathrm{ext}(t, \mathrm{ft}(t)))) \tag{4}$$

Notice that movement is always to the left.

2.3. **Languages.** Trees are coded as terms. Given a lexicon L, the tree language $T(L)$ is the set of all trees that can be generated from L using the operations merge and move. The string language $S(L)$, is the set

$$Y[T(L)] := \{Y(t) : t \in T(L)\} \tag{5}$$

Here, $Y(t)$ is defined inductively as follows.

① If $t = \vec{x}^\frown\vec{y} \in \mathrm{Lex}$ with $\vec{x} \in A^*$ and $\vec{y} \in (CB)^*$ then $Y(t) := \vec{x}$.
② $Y({>}(s, t)) = Y(s)^\frown Y(t)$.
③ $Y({<}(s, t)) = Y(s)^\frown Y(t)$.

3. Coding Structures

We shall code the above structures using modal logic. There will be major changes to the way in which they look, and we shall discuss the ramifications of that later. Our language will be that of modal logic, with several modal operators, $\triangledown_0$ (left standard daughter), $\triangledown_1$ (right standard daughter), $\curlyvee$ and $\curlywedge$. $\curlyvee$ is the head-feature relation and $\curlywedge$ its converse. For a given node x $\curlyvee$ points to the head feature of the constituent headed by x. Finally, we shall add the transitive closure of all of them, which will turn out to also be the transitive closure of just the standard relations. So we write it as $\triangledown^+$. Also,

$$\boxdot\chi := \chi \wedge [\triangledown^+]\chi \tag{6}$$

As usual, although technically we are moving inside a modal language, we take advantage of the fact that **PDL** in full strength is available for us, so we also use its notation. The relations above are actually programs, and for a program α the modality is $[\alpha]$. Also, recall that in certain cases we can make use of the converse, namely when the converse is a partial function. This is guaranteed here for the basic modalities. The set of variables is $\{p_i : i \in \mathbb{N}\}$, and there will be a set of constants (to be determined).

Now we are going to zoom in on the minimalist program. We interpret formulae in so-called **pointed Kripke-frames**. These are pairs $\langle \mathfrak{M}, x\rangle$, where $\mathfrak{M} = \langle M, R, I\rangle$ is a Kripke-frame and $x \in M$. A **Kripke-frame** is a structure $\langle M, R, I\rangle$ where M is a set, R a function from the modalities to M^2 and I a function from the constants to $\wp(M)$. Thus, $R(\triangledown_0), R(\triangledown_1), R(\curlyvee)$, $R(\curlywedge)$ and $R(\triangledown^+)$ are binary relations on M, while the interpretation of the constants are subsets of M.

For the constants we put for every $f \in F$:

$$I''(f) := I(f) \cup I'(f) \tag{7}$$

Next we turn to the encoding of terms. For each $a \in A$ and each $b \in B$ we assume a constant, denoted for simplicity by the element itself. Likewise, there will be constants $\circledast, \oplus, \ominus$ and $\odot$. All these constants are mutually exclusive. This means that we have an axiom of the form

$$\bigwedge \langle f \to \neg f' : f, f' \in F, f \neq f' \rangle \tag{8}$$

where F is the set of constants. We put

$$\mathrm{alph} := \bigvee \langle a : a \in A \rangle \tag{9}$$

Further, for each of these symbols we add the axiom

$$u \to [\triangledown]\bot \tag{10}$$

This makes sure that these constants are true only at leaves. Next we add to F a pair of constants, $\ogreaterthan$ and $\olessthan$. It follows that they are mutually incomparable, and also incomparable with any of the previous. Further, we set

$$\mathrm{lex} := \neg(\ogreaterthan \vee \olessthan) \tag{11}$$

The following shall be an axiom:

$$\mathrm{lex} \to [\triangledown]\,\mathrm{lex} \tag{12}$$

We are now ready for the first step, the encoding of the lexicon. A sequence is generally encoded as an ascending tree. So, if we have the sequence $x_1 x_2 x_3 \cdots$ then this will correspond to the tree with leftmost daughter labelled x_1. The mother of that daughter, n_1, has a right daughter labelled x_2. The mother of that daughter, n_2, has a right daughter labelled x_3, and so on.

We show how to construct the trees for the lexical entries. A single letter $a \in A$ is coded by the structure $\mathfrak{P}_a = \langle \{0\}, R, I \rangle$ such that $R(\Box) := \varnothing$ for all basic modalities $\Box$ and $I(a) := \{0\}$, $I(c) := \varnothing$ for all constants $c \neq a$. Similarly for $\mathfrak{P}_b$, $b \in B$ and $\mathfrak{P}_s$, $s \in \{\oplus, \odot, \ominus, \circledast\}$. So, $\mathfrak{O}(x) := \langle \mathfrak{P}_x, 0 \rangle$. Next, we shall define the code of sb. This shall be based on the frame $\mathfrak{P}_{sb} := \langle \{0, 1, 2\}, R, I \rangle$, where $R(\triangledown_0) := \{\langle 0, 1 \rangle\}$, $R(\triangledown_1) := \{\langle 0, 2 \rangle\}$, $R(\triangledown^+) := \{\langle 0, 1 \rangle, \langle 0, 2 \rangle\}$, $I(b) := \{1\}$, $I(s) := \{0\}$, everything else being assigned to the empty set. $\mathfrak{O}_{sb} := \langle \mathfrak{P}_{sb}, 0 \rangle$. The coding of strings works as follows. Suppose that $\mathfrak{O}(\vec{x}) := \langle \mathfrak{M}, x \rangle$ codes $\vec{x}$ and $\mathfrak{O}(c) := \langle \mathfrak{P}_c, y \rangle$ codes c. Assume for simplicity that the frame have disjoint carrier sets. Then $\mathfrak{O}(\vec{x}^\frown c) :=$

$\langle M \cup N \cup \{*\}, R'', I''\rangle$ where

$$R''(\triangledown_0) := R(\triangledown_0) \cup R'(\triangledown_0) \cup \{\langle *, x\rangle\} \tag{13}$$
$$R''(\triangledown_1) := R(\triangledown_1) \cup R'(\triangledown_1) \cup \{\langle *, y\rangle\}$$
$$R''(\Yup) := \varnothing$$

For the constants we put for every $f \in F$:

$$I''(f) := I(f) \cup I'(f) \tag{14}$$

Next we turn to the encoding of terms. Suppose $\mathfrak{D}(s) = \langle \mathfrak{M}, x\rangle$ encodes the term s and $\mathfrak{D}(t) = \langle \mathfrak{M}', x'\rangle$ the term t. We shall assume that $M \cap M' = \varnothing$ and that they do not contain the symbol $*$. Then $\mathfrak{D}(<(s,t)) = \langle \mathfrak{M}'', x''\rangle$ where $M'' := M \cup M' \cup \{*\}$,

$$R''(\triangledown_0) := R(\triangledown_0) \cup R'(\triangledown_0) \cup \{\langle *, x\rangle\} \tag{15}$$
$$R''(\triangledown_1) := R(\triangledown_1) \cup R'(\triangledown_1) \cup \{\langle *, y\rangle\}$$
$$R''(\Yup) := \varnothing$$

For the constants we put for every $f \in F - \{\olessthan\}$:

$$I''(\olessthan) := I(\olessthan) \cup I'(\olessthan) \cup \{*\} \tag{16}$$
$$I''(f) := I(f) \cup I'(f) \tag{17}$$

$x'' := *$. The code for $>(s,t)$ is the same except for that the sets $I''(\ogreaterthan)$ and $I''(\olessthan)$ are interchanged.

It is perhaps useful to observe that we can define a few formulae that will single out our lexical entries. We define the translation into a modal formula as follows. Put $Y := \{\circledast, \oplus, \ominus, \odot\}$.

$$\mu(a) := a \qquad (a \in A) \tag{18}$$
$$\mu(yb) := \langle \triangledown_0\rangle y \wedge \langle \triangledown_1\rangle b \qquad (y \in Y, b \in B) \tag{19}$$
$$\mu(\vec{x}^\frown u) := \langle \triangledown_0\rangle \mu(x) \wedge \langle \triangledown_1\rangle \mu(u) \qquad (u \in A \cup CB) \tag{20}$$

There will be a few auxiliary constants that we need. First, let

$$\sigma := \langle \triangledown_1; \triangledown_0\rangle(\circledast \vee \ominus \vee \oplus \vee \odot) \tag{21}$$

This constant is true at all nodes that dominate a feature, where by a feature we understand a tree of the form $\mu(x)$, $x \in CB$. It is easy to see that we can write down a formula that exactly codes the structure of any given term.

3.1. **The Lexicon.**

Finally, the code of a lexical entry is this. It will be the code $\langle \mathfrak{M}, x\rangle$ of the sequence $\vec{y} \in L$.

3.2. **Merge and Move.** The principal difference between the present encoding and the one in [3] lies in the fact that derivations do not result in the destruction of labels; rather, they result in the addition of a link. It follows that the structures that they generate are actually different from the structures of minimalism.

Now, when merge or move apply, they use the head-feature. It addition, they shall define a head-feature for the newly created node. Suppose that $\mathfrak{O}(s) := \langle\mathfrak{M}, x\rangle$ is the modal structure associated with s (with x its root), and $\langle\mathfrak{M}', x'\rangle$ the structure associated with t (with x' its root). We shall construct $\mathfrak{O}(\text{merge}(s,t)) := \langle\mathfrak{M}'', x''\rangle$ (if defined). For simplicity we assume both structures to be over disjoint sets and none contains $*$. The new structure is formed over the set $M'' := M \cup M' \cup \{*\}$. The operation is defined if and only if there is $b \in B$ such that

$$(22)\qquad \begin{aligned}\langle\mathfrak{M}, x\rangle &\vDash \langle \curlyvee; \triangledown_1\rangle \odot b\\ \langle\mathfrak{M}', x'\rangle &\vDash \langle \curlyvee; \triangledown_1\rangle \circledast b\end{aligned}$$

In that case, let y be the unique node such that for some z: $x; R(\curlyvee)\ z\ R(\triangledown_0)\ y$.

$$(23)\qquad \begin{aligned}R''(\triangledown_0) &:= R(\triangledown_0) \cup R'(\triangledown_0) \cup \{\langle *, x\rangle\}\\ R''(\triangledown_1) &:= R(\triangledown_1) \cup R'(\triangledown_1) \cup \{\langle *, x'\rangle\}\\ R''(\curlyvee) &:= R(\curlyvee) \cup R'(\curlyvee) \cup \{\langle *, y\rangle\}\end{aligned}$$

For $f \in F - \{\circledless\}$:

$$(24)\qquad \begin{aligned}I''(\circledless) &:= I(\circledless) \cup I'(\circledless) \cup \{*\}\\ I''(f) &:= I(f) \cup I'(f)\end{aligned}$$

Finally, $x'' := *$. This finishes the definition of $\mathfrak{O}(\text{merge}(s,t))$ for this case. It is also defined if there is $b \in B$ such that

$$(25)\qquad \begin{aligned}\langle\mathfrak{M}, x\rangle &\vDash \langle \curlyvee; \triangledown_1\rangle \circledast b\\ \langle\mathfrak{M}', x'\rangle &\vDash \langle \curlyvee; \triangledown_1\rangle \odot b\end{aligned}$$

In that case, y will be chosen such that $x'\ R'(\curlyvee)\ z\ R'(\triangledown_0)\ y$, and the constants are defined thus. For $f \in F - \{\circledgtr\}$:

$$(26)\qquad \begin{aligned}I''(\circledgtr) &:= I(\circledgtr) \cup I'(\circledgtr) \cup \{*\}\\ I''(f) &:= I(f) \cup I'(f)\end{aligned}$$

Next we come to move. Suppose that $\mathfrak{O}(s) = \langle\mathfrak{M}, x\rangle$. Suppose further that $\langle\mathfrak{M}, x\rangle \vDash \langle \curlyvee; \triangledown_1\rangle \oplus b$ for some b. Then $\mathfrak{O}(\text{move}(s))$ will be defined just in case there is a unique maximal y such that $\langle\mathfrak{M}, y\rangle \vDash \langle \curlyvee; \triangledown_1\rangle \ominus b$. We fix this y. Assume that $* \notin M$. Put $\mathfrak{O}(\text{move}(s)) := \langle\mathfrak{M}', *\rangle$. Furthermore, let u

be the unique node such that there is a v and $x\ R(\curlyvee)\ v\ R(\triangledown_0)\ u$.

(27)
$$\begin{aligned} R''(\triangledown_0) &:= R(\triangledown_0) \cup \{\langle *, y\rangle\} \\ R''(\triangledown_1) &:= R(\triangledown_1) \cup \{\langle *, x\rangle\} \\ R''(\curlyvee) &:= R(\curlyvee) \cup \{\langle *, u\rangle\} \end{aligned}$$

For the constants: For $f \in F - \{\circleddash\}$ put

(28)
$$\begin{aligned} I''(\circleddash) &:= I(\circleddash) \cup I'(\circleddash) \cup \{*\} \\ I''(f) &:= I(f) \cup I'(f) \end{aligned}$$

In movement, the site is added on the left and the head is always to the right.

4. Axiomatisation

4.1. **The Basic System.** The basic axiom system is that axiomatises the class of structures with the following properties:

Proposition 1. *The structures $\mathfrak{D}(s)$ for a term s have the following properties:*

(1) $\langle M, R(\triangledown_0), R(\triangledown_1)\rangle$ *is a binary branching tree;*
(2) $R(\triangledown^+) = (R(\triangledown_0) \cup R(\triangledown_1))^+$*;*
(3) $R(\curlyvee) \subseteq R(\triangledown^+)$*;*
(4) *If $x\ R(\curlyvee)\ y$ then y has no $R(\curlyvee)$-successor.*
(5) $R(\curlywedge)$ *is the converse of* $R(\curlyvee)$*.*
(6) *the relations $R(\triangledown_0)$, $R(\triangledown_1)$ and $R(\curlyvee)$, and their converses are partial functions;*
(7) *x has a successor via $R(\triangledown_0)$ iff it has a successor via $R(\triangledown_1)$.*
(8) *x has a $R(\curlyvee)$-successor iff x is not lexical;*
(9) *If x has a $R(\curlywedge)$-successor, then x is lexical and if $y\ R(\triangledown_0)\ x$ is lexical then y has a $R(\curlywedge)$-successor too.*

The proof is by induction on s. Perhaps the least straightforward part is the fact that $R(\curlyvee)$ and its converse are partical functions. It is clear by the definition of $R(\curlyvee)$ that it is a partial function. At each step, the newly added root is linked to exactly one node via $R(\curlyvee)$. Now, to see that also the converse $R(\curlywedge) := R(\curlyvee)^{\smile}$ is a partial function, let us start with a lexical entry. There it is clear by construction. Notice that the root is self-accessible via $R(\curlywedge)$. By induction we verify that for every skeletal node x (ie a node satisfying σ): if there is $R(\curlywedge)$-successor v, and if $y\ R(\triangledown_0)\ x$ also is skeletal then y has a $R(\curlywedge)$-successor w and moreover $v\ R(\triangledown^+)\ w$. So, as we go up the skeleton then $R(\curlywedge)$-successors move down. (This follows in the Stabler framework from the process of 'nibbling off' features.) Hence, when we merge two structures or when we move it is the skeletal daughter of the head of the constituents that is $R(\curlywedge)$-related to the new root. This node has not yet been assigned any successor.

Proposition 2. *The logic of structures satisfying the properties listed in Proposition 1 is axiomatisable over* K_6 *by the following axioms (with* $[\triangledown]\chi := [\triangledown_0]\chi \wedge [\triangledown_1]\chi$*).*

① $\Diamond p \to \Box p$, $\Diamond \in \{\langle\triangledown_0\rangle, \langle\triangledown_1\rangle, \langle\curlyvee\rangle, \langle\curlywedge\rangle\}$;
② $\Diamond p \to \langle\triangledown^+\rangle p$, $\Diamond \in \{\langle\triangledown_0\rangle, \langle\triangledown_1\rangle, \langle\curlyvee\rangle, \langle\curlywedge\rangle\}$;
③ $[\triangledown^+]([\triangledown^+]p \to p) \to [\triangledown^+]p$;
④ $[\triangledown]p \wedge [\triangledown^+](p \to [\triangledown]p) \to [\triangledown^+]p$;
⑤ $p \to [\curlyvee]\langle\curlywedge\rangle p$;
⑥ $p \to [\curlywedge]\langle\curlyvee\rangle p$;
⑦ $[\curlyvee^2]\bot$;
⑧ $\neg\,\mathrm{lex} \leftrightarrow \langle\curlyvee\rangle\top$;
⑨ $\mathrm{lex} \wedge \langle\triangledown_0;\curlywedge\rangle\top \to \langle\curlywedge\rangle\top$.

Notice that the class of structures for this logic is slightly wider than the one admitted by Proposition 1. It is admitted that isomorphic constituents are collapsed into a single constituent. This is however a harmless complication.

4.2. Merge and Move.

In particular, we need to establish the peculiar structure that the operations merge and move establish. Let us first notice that for each node x that is not lexical we can read off whether or not it has been formed through merge or through move. It has been formed through left-headed merge if:

$$\langle\mathfrak{M}, x\rangle \vDash \chi_\ell := \bigvee_{b\in B} \langle\triangledown_0;\curlyvee\rangle \circledast b \wedge \langle\triangledown_1;\curlyvee\rangle \odot b \tag{29}$$

It has been formed through right-headed merge if

$$\langle\mathfrak{M}, x\rangle \vDash \chi_r := \bigvee_{b\in B} \langle\triangledown_1;\curlyvee\rangle \circledast b \wedge \langle\triangledown_0;\curlyvee\rangle \odot b \tag{30}$$

It has been formed through move iff

$$\langle\mathfrak{M}, x\rangle \vDash \chi_m := \bigvee_{b\in B} \langle\triangledown_1;\curlyvee\rangle \oplus b \wedge \langle\triangledown_0;\curlyvee\rangle \ominus b \tag{31}$$

To make sure that each intermediate node has been formed in this way we issue the following axiom:

$$\neg\,\mathrm{lex} \to \chi_\ell \vee \chi_r \vee \chi_m \tag{32}$$

We shall have to write axioms for these three cases. For left-headed merge the formula is this.

$$\chi_\ell \to (\langle\curlyvee\rangle p \leftrightarrow \langle\triangledown_0;\curlyvee;\triangledown_0\rangle p) \tag{33}$$

This says that the head-feature of the root is equal to the $R(\triangledown_0)$-daughter of the head-feature of the left-hand daughter. For right-headed merge we do this:

(34) $$\chi_r \to (\langle\curlyvee\rangle p \leftrightarrow \langle\triangledown_1;\curlyvee;\triangledown_0\rangle p)$$

This leaves us to account for movement. Before we can begin to define it, we need a few tools. Let

(35) $$\mathrm{con}(p) := [\triangledown^*](p \to [\triangledown]p) \wedge [\triangledown^*](\neg p \to \neg(\langle\triangledown_0\rangle p \wedge \langle\triangledown_1\rangle p))$$

It is easy to see that if $\mathrm{con}(p)$ is true at a the root, then the set of values of p is a constituent. Let us pick a node x and assume that the variable p is true at the constituent headed by y, where y is the right hand daughter of x. We can achieve this if we have

(36) $$x \vDash \neg p; \mathrm{con}(p); \langle\triangledown_1\rangle p$$

Then the formula $\langle\curlywedge\rangle p$ is true at all nodes that contain a checked feature as last element within the constituent headed by y. The head-daughter of x is not of that form:

(37) $$x \vDash \langle\curlyvee\rangle\neg\langle\curlywedge\rangle p$$

We should also encode the fact that the head daughter chooses the nearest constituent that has an unchecked feature. This can be said as follows.

(38) $$\langle\curlyvee\rangle q \to \langle((\neg\,\mathrm{lex} \vee \sigma \wedge \langle\curlywedge\rangle p); \triangledown)^*\rangle q$$

This formula declares that if q is true at least at the head daughter of x then we can reach q by going down all nonlexical nodes or all lexical nodes that do not contain unchecked features. Let

(39) $$\alpha(b) := ((\neg\,\mathrm{lex}?; \triangledown) \cup (\sigma \wedge \langle\curlywedge\rangle p \wedge \neg\langle\triangledown_1\rangle \ominus b)?; \triangledown_0)^*; \\ (\langle\triangledown_1\rangle \ominus b \wedge \neg\langle\curlywedge\rangle p)?$$

So, the formula we seek is

(40) $$\langle\triangledown_1;\curlyvee\rangle \oplus b \wedge \mathrm{con}(p) \wedge \neg p \wedge \langle\triangledown_1\rangle p \wedge \langle\curlyvee\rangle q \to \langle\alpha(b)\rangle q$$

Notice that the formula does not imply that the element we are looking for is unique. For clearly, the nodes that can be reached following only checked features until we reach an unchecked feature must be incomparable but if they are, nothing contradicts the formula. To ensure uniqueness, we add this axiom:

(41) $$\langle\triangledown_1;\curlyvee\rangle \oplus b \wedge \mathrm{con}(p) \wedge \neg p \wedge \langle\triangledown_1\rangle p \wedge \langle\alpha(b)\rangle q \to [\alpha(b)]q$$

Definition 3. $\mathsf{Min}(A, B)$ *is the logic obtained by adding to the logic defined in Proposition 2 the axioms* (8)*,* (10)*,* (12)*,* (32)*,* (33)*,* (34)*,* (40)*, and* (41)*.*

4.3. **Structures à la Stabler.** We need to address the fact that we generate structures that are different. The structures defined here leave a lot of material intact; it is not removed. The question that arises is whether we can actually find the generated Stabler-trees inside our structures. This is the case. Let $\langle M, R, I\rangle$ be a structure in our sense with root x. Notice first that the relations $R(\triangledown_0)$ and $R(\triangledown_1)$ define a tree, and the Stabler-trees are simply a subtree. We need to skip all the empty material. A skeletal node x is **checked** if $x \vDash \langle\curlywedge\rangle\top$. This means that the feature of this node has been checked off. These nodes (and their $R(\triangledown_1)$-daughter constituents) must be skipped. However, this is not all that needs to be skipped. Basically, everything that is below a node that has been moved needs to be skipped. For that, call a constituent below a node satisfying $\langle\triangledown_1\rangle \ominus b \wedge \langle\curlywedge\rangle\top$ **displaced**. Now, the displaced node is really moved in a Stabler-structure, while here we just add links. Now, the idea is that it is pronounced only if one is going down using $R(\triangledown_0)$-moves.

This is not easy to eliminate empty stuff. Again, a lot of details need to be considered. The best approach is this. We are changing the structures slightly in the following way. In the move step, we actually copy the moved constituent, and mark the lower copy as empty. Let's perform this operation. In the beginning, $I(\varnothing)$ and merge will not add any empty nodes. However, let's look again at the move-step.

Suppose that $\mathfrak{O}(s) = \langle\mathfrak{M}, x\rangle$. Suppose further that $\langle\mathfrak{M}, x\rangle \vDash \langle\curlyvee; \triangledown_1\rangle \oplus b$ for some b. Then $\mathfrak{O}(\text{move}(s))$ will be defined just in case there is a unique maximal y such that $\langle\mathfrak{M}, y\rangle \vDash \langle\curlyvee; \triangledown_1\rangle \ominus b$. We produce a copy $\langle\mathfrak{M}', y'\rangle$ which is isomorphic to $\langle\mathfrak{M}, y\rangle$ (in particular, y' is the root of $\mathfrak{M}'$, as y is the root of $\mathfrak{M}$). Assume that $* \notin M$. Put $\mathfrak{O}(\text{move}(s)) := \langle\mathfrak{M}'', *\rangle$. Furthermore, let u be the unique node such that there is a v and $x\ R(\curlyvee)\ v\ R(\triangledown_0)\ u$.

$$\begin{aligned} R''(\triangledown_0) &:= R(\triangledown_0) \cup R'(\triangledown_0) \cup \{\langle *, y'\rangle\} \\ R''(\triangledown_1) &:= R(\triangledown_1) \cup R'(\triangledown_1) \cup \{\langle *, x\rangle\} \\ R''(\curlyvee) &:= R(\curlyvee) \cup R'(\curlyvee) \cup \{\langle *, u\rangle\} \end{aligned} \tag{42}$$

For the constants: For $f \in F - \{\circledtriangleright\}$ put

$$\begin{aligned} I''(\circledtriangleright) &:= I(\circledtriangleright) \cup I'(\circledtriangleright) \cup \{*\} \\ I''(f) &:= I(f) \cup I'(f) \end{aligned} \tag{43}$$

In movement, the site is added on the left and the head is always to the right.

The axiomatisation of these structures is not different in that the $R(\curlyvee)$-daughter of the constituent is not reachable via $R(\triangledown_0)$. This is an axiom that we have not written down yet, so we let matters stand as they are. The logic is $\mathsf{Mov}(A, B)$. Instead, we refer to Part I for details on how to enforce a true tree structure on $\langle M, R(\triangledown_0), R(\triangledown_1)\rangle$. One way is to add converses $\vartriangle_0$ and $\vartriangle_1$, and add the following axioms.

(44) $$\langle \vartriangle_0 \rangle p \to [\vartriangle_0] p$$

(45) $$\langle \vartriangle_1 \rangle p \to [\vartriangle_1] p$$

(46) $$\langle \vartriangle_0 \rangle \top \to [\vartriangle_1] \bot$$

(47) $$\langle \vartriangle_1 \rangle \top \to [\vartriangle_0] \bot$$

This turns them into real trees. I have noted however that the first two are not needed (if the relations are not functions they point to structure that we do not need) and the last two are constant, so we can safely ignore all of this for decidability.

Now let us first see how we can define those nodes that exist in a Stabler-tree. These are the nonempty nodes and the nodes such that there is no node $u \vDash \langle \curlyvee; \triangledown_1 \rangle \ominus b$ above them.

(48) $$\xi := \bigvee_{a \in A} a \vee \bigvee_{a \in A} \langle \triangledown \rangle a$$

So, let m be a variable and put

(49) $$\begin{array}{rl} d(m) := & \neg m \\ \wedge & \boxdot(m \to \boxdot m) \\ \wedge & \boxdot(\neg m \wedge (\chi_\ell \vee \chi_r) \to [\triangledown]\neg m) \\ \wedge & \boxdot(\neg m \wedge \chi_m \to [\triangledown]\neg m \wedge [\triangledown_1; \curlyvee; \triangledown^*] m) \\ \wedge & \boxdot(\sigma \wedge \neg m \to [\triangledown^*]\neg m) \\ \wedge & \boxdot(\xi \wedge \neg m \to [\triangledown]\neg m) \end{array}$$

Lemma 4. *Let $\langle \mathfrak{M}, x \rangle$ be a tree wrt $R(\triangledown)$ and assume that $\langle \mathfrak{M}, \beta, x \rangle \vDash d(m)$. Then $\beta(m)$ is the set of displaced constituents.*

Proof. The clauses are correct. They also lead to a unique assignment. To see this, take a node and assume that all the nodes y such that $y\ R(\triangledown^+)\ x$ have a unique assignment. Then show that x has unique assignment. This is clear if the mother is lexical or has assignment m. If it is nonlexical and $\neg m$, it is either χ_ℓ, χ_r or χ_m. All these cases are cared for. Thus, after uniqueness we see the correctness. We do induction over the depth of a node. The root is not displaced, as required. The daughters of a displaced constituent member is itself a member of a displaced constituent. Next, the daughters of a merged node are not displaced provided that the node itself is not displaced. If x is lexical and not in a displaced constituent, so is the entire constituent that it heads. This leaves only a few cases to consider. Next, when we consider a constituent formed through movement, there are three types of daughters to be considered. Empty is the $R(\curlyvee)$-daughter constituent, while the $R(\triangledown_i)$-daughters are not displaced. In the latter case we cannot yet say that the entire constituent is of that kind. □

Next set $p(n)$ to be

$$
(50)\qquad
\begin{array}{rll}
p(n) := & & \neg n \\
& \wedge & \boxdot(\langle\curlywedge\rangle\top \to n) \\
& \wedge & \boxdot(n \to [\triangledown_1; \triangledown^*]n) \\
& \wedge & \boxdot(\neg\,\mathrm{lex} \to [\triangledown]\neg n) \\
& \wedge & \boxdot(\xi \to \neg n) \\
& \wedge & \boxdot(\neg\xi \wedge \sigma \wedge [\curlywedge]\bot \to (\neg n \wedge [\triangledown_1; \triangledown^*]\neg n)
\end{array}
$$

Lemma 5. *Let $\langle\mathfrak{M}, x\rangle$ be a tree wrt $R(\triangledown)$ and assume that $\langle\mathfrak{M}, \beta, x\rangle \vDash p(n)$. Then $\beta(n)$ is the set of empty nodes.*

Finally, given that the root satisfies $d(m)$ and $p(n)$, the set of nonempty nodes is defined by

$$(51)\qquad \upsilon := \neg m \wedge \neg n$$

$$(52)\qquad N := \{y : \langle\mathfrak{M}, \beta, y\rangle \vDash \upsilon\}$$

We shall define the new relations as follows. $x\ S(\triangledown_0)\ y$ shall hold if either $x\ R(\triangledown_0)\ y$ and $y \vDash \neg\,\mathrm{lex} \vee \xi$, or else if $y \vDash \sigma$ then y is the largest σ-node below x such that $x \in N$. $x\ S(\triangledown_1)\ y$ iff $x\ R(\triangledown_1)\ y$. Likewise, $x\ S(\curlyvee)\ y$ iff $x\ R(\curlyvee)\ y$. $J(c) := N \cap I(c)$. This defines the structure $\mathrm{Stb}(\mathfrak{M})$. Now we define the Stabler-transform of a formula as follows:

$$
(53)\qquad
\begin{array}{ll}
p^\dagger & := p \wedge \upsilon \\
(\neg\chi)^\dagger & := \neg(\chi^\dagger) \wedge \upsilon \\
(\chi \wedge \chi')^\dagger & := \chi^\dagger \wedge \chi'^\dagger \\
(\langle\triangledown_1\rangle\chi)^\dagger & := \langle\triangledown_1; \upsilon?\rangle\chi^\dagger) \\
(\langle\curlyvee\rangle\chi)^\dagger & := \langle\curlyvee\rangle; \upsilon?\rangle\chi^\dagger) \\
(\langle\triangledown_0\rangle\chi)^\dagger & := \langle\triangledown_0; (\neg\upsilon?; \triangledown_0); \upsilon?\rangle\chi^\dagger
\end{array}
$$

The Stabler transform allows to interpret the structures of the Stabler-structures in the structures from which they are "stripped". Notice that this translation extends to all the other PDL-constructs, so every PDL-query for the Stabler-trees can be translated into a PDL-query about our structures. What connects the two is the following

Theorem 6. $\langle\mathrm{Stb}(\mathfrak{M}), x\rangle \vDash \chi$ *iff* $\langle\mathfrak{M}, x\rangle \vDash d(m) \wedge p(n) \to \chi^\dagger$

The proof is done by induction on the structure of χ and is routine.

4.4. **Languages.** Let $L \subseteq A^*(CB)^*$ be a lexicon and $\mathcal{K}$ the class of frames associated with the words of the lexicon. Set

$$(54)\qquad \lambda := \bigvee\langle\mu(\vec{x}) : \vec{x} \text{ is a prefix of some } \vec{y} \in L\rangle$$

Then the class of structures created by L is the one that satisfies

$$(55)\qquad \mathrm{lex} \to \lambda$$

Notice that this is a constant formula. This means that everything is just a matter of showing the base logic decidable. Everything else follows right away using the fact that constant axioms preserve decidability and complexity.

Now, suppose we want to know whether a certain sentence is grammatical. For that we have to ask whether it is the yield of a complete tree. Let c be a basic feature. A tree is **complete** iff there is exactly one occurrence of an unchecked feature in T, which happens to be the head feature, and it is $\circledast\,\mathsf{c}$. We use the elsewhere modality

(56) $$[\neq]\chi := [\triangledown^{+} \cup (\vartriangle^{*}; \vartriangle_0; \triangledown_1; \triangledown^{*}) \cup (\vartriangle^{*}; \vartriangle_1; \triangledown_0; \triangledown^{*})]\chi$$

Now put

(57) $$\kappa := \langle\triangledown_0; \triangledown_1\rangle(\circledast\,\mathsf{c} \wedge [\neq; \curlywedge]\bot \vee \vee\langle\triangledown_1; \triangledown_0\rangle(\circledast\,\mathsf{c} \wedge [\neq; \curlywedge]\bot$$

It then turns out that $\langle\mathfrak{M}, x\rangle$ is complete iff

(58) $$\langle\mathfrak{M}, x\rangle \vDash \kappa$$

We remark that the addition of converse modalities introduces variables into the formula; however, their value is completely fixed. Technically, we have to read the above as saying: for all valuations for the variables the formula contains the above holds good. Computationally, this does not increase the complexity since there is only one valuation that satisfies the antecedent that fixes the valuation for the variables, and computing this valuation is done in linear time.

5. Decidability

In view of the previous discussion it is apparent that we have to solve only the decidability of the basic logic $\mathsf{Mov}(A, B)$, probably with the addition of the converses so at to be able to express the fact that the structures are really trees. Moreover, we may remove any constant formulae that blurs the view.

We shall retrace the proof of decidability of Part II. Details will be only sketched if they have been proved already in that paper.

The first issue we shall have to take up is that of the upward looking modalities. There is a way to eliminate them, but this elmination process turns out to be circular. (It yields an implicit definition, which is why we cannot eliminate them directly. But we can effectively do an elimination based on implicit definitions.) Thus, we propose the following method.

Let Δ be a set of formulae closed under subformulae. Let $A(\Delta)$ be the set of Δ-atoms. Let $H(\Delta)$ be the set of nonrepeating sequences over $A(\Delta)$. For every $h \in H$ we introduce variables p_h and q_h plus the the following formulae:

$$
\begin{aligned}
X(\Delta) :=&\{p_h \to [\triangledown]p_h : h \in H\}\\
&\cup \{q_h \to \langle\triangledown^*\rangle p_h : h \in H\}\\
&\cup \{p_h \wedge \langle\triangledown^+\rangle a \to [\triangledown]\neg p_{h;a} : h; a \in H\}\\
&\cup \{p_h \wedge \langle\triangledown_0; \triangledown^*\rangle a \to [\triangledown_1; \triangledown^*]\neg q_{h;a} : h; a \in H\}\\
&\cup \{p_h \wedge a \wedge q_{h;a} \wedge [\triangledown^+]\neg a \to p_{h;a} : h; a \in H\}
\end{aligned}
\tag{59}
$$

Suppose $\langle\mathfrak{M}, \beta, x\rangle \vDash \boxdot X(\Delta)$ valuation such that $\langle\mathfrak{M}, \beta, x\rangle \vDash [\triangledown^*]X(\Delta); p_\varnothing$. Then the set $\beta(p_h)$ is a constituent for every $h \in H$. Moreover, it is the constituent headed by the egregious point with address h.

The construction now proceeds as follows. We shall require a model for $\varphi; \boxdot X(_(\varphi); \boxdot\Pi(\varphi)$ where $\Pi(\varphi)$ consists of instances of the axioms where in place of p we have inserted all possible p_h, $h \in H$. Now we create the model of all egregious points. It satisfies φ, and the variables p_h now identify all possible constituents of this new model. It needs to be shown that the newly defined relations establish a structure for the logic.

(To be continued.)

References

[1] Marcus Kracht. Is there a genuine modal perspective on feature structures? *Linguistics and Philosophy*, 18:401 – 458, 1995.

[2] Marcus Kracht. Logic and Syntax – A Personal Perspective. In Maarten de Rijke, Krister Segerberg, Heinrich Wansing, and Michael Zakharyaschev, editors, *Advances in Modal Logic '98*, pages 337 – 366. CSLI, 2001.

[3] Edward P. Stabler. Derivational Minimalism. In Christian Retoré, editor, *Logical Aspects of Computational Linguistics (LACL '96)*, number 1328 in Lecture Notes in Artificial Intelligence, pages 68 – 95, Heidelberg, 1997. Springer.

Department of Linguistics, UCLA, 3125 Campbell Hall, PO Box 951543, Los Angeles, CA 90095-1543, `kracht@humnet.ucla.edu`