Tree Logic: A Formal Perspective on Transformations

Herbert Ruessink*, STT/OTS Utrecht University

Abstract

A formal framework is required to investigate the interaction between transformations and constraints with respect to the manner in which linguistic structures are described in (Chomsky, 1992). In this paper I will present the outline of one possible instantiation of such a formal framework. The formalism, Tree Logic, is based on First Order Predicate Logic and adopts a standard model-theoretic semantics. Tree Logic is interpreted in the domain of tree structure interpretations. Tree Logic is defined to a large extent analogously to Smolka's Feature Logic (Smolka, 1989).

1 Introduction

Linguistic theories in the Chomskyan tradition (eg. "Government and Binding Theory" and "The Minimalist Program") employ several levels of representation and each linguistic utterance is associated with more than one (tree) structure. This approach requires that the linguistic theories contain components to perform two general tasks: (i) to describe which tree structures are well-formed at some level of representation; and (ii) to describe which tuples of (tree) structures together form a well-formed representation of a single linguistic utterance. The first task can be regarded as well-formedness conditions on structures and the second can be regarded as wellformedness conditions on the relation between structures.

In the earlier versions of the Chomskyan framework these two tasks were performed by a single derivational component that consisted of a context-free base and a set of (ordered) transformational rules. Since the early seventies the power of the transformational component has been reduced and an additional constraints system was introduced to describe well-formed structures at the levels of representation. However, even in the most recent development in the Chomskyan framework (the "Minimalist Program" (Chomsky, 1992)) the transformational component also contributes to determining the well-formedness of the structures at the levels of representation: the elimination of D-structure implies that the transformations that apply before the

*This researched described in this paper was supported by the CEC project LRE–61–061 "The Reusability of Grammatical Resources"

SPELL-OUT mapping to PF only serve to determine admissable structures at PF and LF.

This state of affairs is a less elegant property of the Chomskyan tradition: one component contributes to two tasks and one task is performed by two components. Posing two components (transformations and filters) to perform a single task does not agree with the declared aim of simplicity and the complex interaction between the transformation and the constraints does not present an clear insight into the properties of the structures at the representational levels.

Two related questions arise from this observation. Firstly, is it possible to separate the transformational component into one part that describes admissable tuples and another that describes admissable structures? Secondly, is it possible to unify the second part within the general framework of well-formedness constraints? It is clear that these questions can only be investigated within the context of a formal framework.

In this paper I will present the outline of one possible instantiation of the formal framework that is required. The formalism, Tree Logic, is based on First Order Predicate Logic and adopts a standard model-theoretic semantics. Tree Logic is interpreted in the domain of tree structure interpretations. Tree Logic is defined to a large extent analogously to Smolka's Feature Logic (Smolka, 1989). Tree Logic is not a formalisation of the Minimalist Program or GB, just as a feature logic is not a formalisation of HPSG. Tree Logic provides a framework within which Chomskyan theories can be expressed. There exist alternatives to taking trees as the structural representation of GB or the Minimalist Program. Ruessink (1994) investigates structures that are based on the phrase-marker relations containment and inclusion.

In section 2 I will briefly introduce the "Minimalist Program" and illustrate the manner in which the transformations in the overt syntax component determine properties of admissable structures. In section 3 I will present Tree Logic. Section 4 presents two brief examples of the way in which Tree Logic can be used to express linguistic notions and constraints. The final section contains some concluding remarks.

2 Transformations in the Minimalist Program

In (Chomsky, 1992), Chomsky presents a theory in which only the interface levels (PF and LF) constitute representational levels. This section briefly summarises a part of his proposal to show how restrictions of the derivation constrain the structures that are acceptable at the interface levels.

2.1 Outline of the Minimalist Program

The basic structures of Chomsky's proposed approach are derived from elements taken from the lexicon. Any item α in the lexicon can be freely projected to one of the following $\bar{\mathrm{X}}$ structures:

1. (a) $[_{\alpha}\ \alpha]$

   (b) $[_{\alpha^1} [_{\alpha}\ \alpha]]$

(c) $[_{\alpha^2}[_{\alpha^1}[_{\alpha}\ \alpha]]]$

A set of such structures forms the base of the transformational operations. The binary operation GT combines two structures whereas the unary operation Move-α maps a single structure to another. A derivation is represented as sequences of operations operating on the output of previous operations. At any stage the operation SPELL-OUT may be applied to a single resulting structure. This will map the structure to PF. The derivation then proceeds to LF, under the restriction that the lexicon no longer can be accessed. This effectively restricts the derivation to applications of Move-α.

The transformations GT and Move-α are described as follows. The generalised transformation GT takes a phrase-marker K, adds an empty node ϕ somewhere in the structure and replaces ϕ by a second phrase-marker K'. The second operation, Move-α, maps a single phrase-marker K into a phrase-marker K'. It operates similarly to GT: it creates an empty position ϕ and substitutes an element α from K for ϕ, resulting in K^*. This operation leaves behind a trace t for α and forms the chain (α, t). The structures K^* that are produced by the transformations must meet the constraints of $\bar{\mathrm{X}}$-Theory, which is extended to cover adjunction structures:

2. (a) $[_X\ Y\ X]$
   (b) $[_{\bar{\bar{\mathrm{X}}}}\ \bar{\bar{\mathrm{Y}}}\ \bar{\bar{\mathrm{X}}}]$

The two identically labeled nodes in an adjunction structure, eg. the two occurrences of $\bar{\bar{\mathrm{X}}}$ in 2b, are taken to be two components or *segments* of a single unit, the category $\bar{\bar{\mathrm{X}}}$. Only those categories that are dominated by all segments of a category are *dominated* by that category. A category that is dominated by some segment is *contained* by the category to which the segment belongs, and a category that is dominated by no segment is *excluded*.

Substitution and adjunction by GT and sc move-α are constrained in overt syntax (ie. before the operation SPELL-OUT) by the fact that that ϕ must be external to the targeted phrase-marker K. In other words, adding ϕ to a structure X^1 results in $[_\beta\ \phi\ X^1]$. The empty element can be replaced by either a different structure Y or by an element α from X^1. Since this is subject to $\bar{\mathrm{X}}$-Theory, both Y and α must be maximal projections and $\beta = X^2$.

2.1.1 The Role of Transformations

The role of the transformational component in overt syntax can only be that of describing admissable structures. With the elimination of D-structure only PF and LF remain. LF is derived after SPELL-OUT has applied and the transformations that apply before that point do not contribute to constraining the relation between PF and LF.

In addition, the restriction that substitution extends the targeted phrase-marker serves to rule out certain structures as admissable representations at the interface level. (Chomsky, 1992) indicates two consequences of this restriction. The first is that it expresses the condition of the strict cycle. Take for instance the following example where superraising applies to derive 3b from 3a. Subsequently *it* is inserted, resulting in 4. The first step does not violate relativised minimality (Rizzi, 1990) as the intervening phrase is inserted later. However, this insertion is prohibited by the restriction on substitution. Consequently, example 4 is ruled out because it cannot be the result of a correct derivation.

3. (a) $[_{I^1}$ seems $[_{I^1}$ to be certain [John to be here]]]
   (b) $[_{I^2}$ John seems $[_{I^1}$ to be certain [t to be here]]]

4. John seems it to be certain to be here

The second consequence is that there cannot be any raising to complement position. A structure of the form $[_{X^1}\ X\ YP]$ cannot result in a structure $[_{X^1}\ X\ ZP\ YP]$, irrespective of whether ZP is raised from within YP or inserted by GT.

3 Tree Logic

In this section I present a framework for the formalisation of linguistic theories in the Chomskyan tradition. The framework is a logic for tree structures, called Tree Logic.

3.1 Labeled and Ordered Trees

The structures that are used to represent linguistic objects in the Chomskyan framework are phrase markers. Phrase markers are none other than labeled, ordered trees. These structures encode two kinds of information: configurational and categorial. Configurational information is expressed in terms of the dominance and precedence relations. Categorial information is represented by the labels associated with the (roots of) the trees. For the representation of categorial information I will assume sets Lab of labels such as $\{n, v, p, \ldots, bar0, bar1, bar2, \ldots\}$. The symbols $l, l_1, l_2, \ldots$ will denote labels. I also assume a set N of nodes. I will assume 5 as the definition of a tree structure[1].

5. A tree T is a 6-tuple $T = \langle N_T, D_T, P_T, r_T, L_T, Lab_T \rangle$ where

   $N_T \subseteq N$, a finite set of nodes of T;

   $D_T \subseteq N_T \times N_T$, the reflexive, antisymmetric, transitive dominance relation;

   $P_T \subseteq N_T \times N_T$, the reflexive, asymmetric, transitive precedence relation;

   $r_T \in N_T$, the root of T;

   $L_T \subseteq N_T \times Lab$, the labeling relation of T;

   Lab_T is a set of atomic labels

[1] This definition is a slightly modified version of the definition given in (Wall, 1972).

such that for all a, b, c and $d \in N_T$ the following hold:

(i) The Single Root Condition: $(r, a) \in D_T$

(ii) The Exclusivity Condition: $((a, b) \in D_T \vee (b, a) \in D_T) \leftrightarrow \neg((a, b) \in P_T \vee (b, a) \in P_T)$

(iii) The Non-tangling Condition: $((a, b) \in P_T \wedge (a, c) \in D_T \wedge (b, d) \in D_T) \rightarrow (c, d) \in P_T$

I will assume that labels represent various properties and that these properties are not always mutually exclusive. The fact that a node x in a tree T is an NP might for instance be represented by "$(x, noun), (x, bar2) \in L_T$."

The symbol $\downarrow(T, n)$ will refer to the subtree T' of T rooted in the node n:

6. $\downarrow(T, n)$ is the subtree $T' = \langle N_{T'}, D_{T'}, P_{T'}, n, L_{T'} \rangle$ of the tree $T = \langle N_T, D_T, P_T, r_T, L_T \rangle$ where

   (a) $N_{T'} = \{x \in N_T | (n, x) \in D_T\}$

   (b) $\forall x, y \in N_{T'}\ (x, y) \in D_{T'} \leftrightarrow (x, y) \in D_T$

   (c) $\forall x, y \in N_{T'}\ (x, y) \in P_{T'} \leftrightarrow (x, y) \in P_T$

   (d) $\forall x \in N_{T'}\ (x, y) \in L_{T'} \leftrightarrow (x, y) \in L_T$

The subtree $\downarrow(T, n)$ is defined by taking the set of nodes dominated by n in T as the set $N_{T'}$ and by retaining the dominance, precedence and labeling relations that hold for these nodes in T. I will also employ the subtree relation between trees T and T', which holds if T' is a subtree of T. Subtrees are defined on the basis of the dominance relation and the subtree relation therefore is reflexive, antisymmetric and transitive. An immediate subtree is rooted in a node that is immediately dominated by the root of the larger tree.

Finally, T_{NL} will refer to the set of all trees such that for each element T of this set $N_T \subseteq V$ and $Lab_T = L$, and t_{NL} will be any subset of this set.

3.2 Interpretations

Tree Logic is intended to be able to model (derivational) relations between trees rather than just relations between nodes. It differs in this respect from the formalisations of tree descriptions presented by (Rogers and Vijay-Shanker, 1992) and (Blackburn *et al.*, 1993). The domain of interpretation will therefore be a set of trees rather than a set of nodes. This means that the precedence, dominance and labeling relations will be represented as relations defined on trees, rather than relations defined on nodes. The obvious step in determining the semantics of the formalism is taking a set of trees as the domain of interpretation and deriving the interpretations of predicates from the relations that exist within the structures. I will call such interpretations "tree structure interpretations." This is similar to the approach to Feature Logic in (Smolka, 1989).

I assume the set V of variables and a set P of predicate symbols. Let p stand for any predicate symbol, and let Ar be the arity function $P \mapsto N^+$. The set of predicate symbols contains at least the predicates Dom and $Prec$ and every label l in the set of labels L.

3.2.1 Tree Structure Interpretations

A tree structure interpretation will be a pair $(U^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where $U^{\mathcal{I}}$ is a set of trees and $\cdot^{\mathcal{I}}$ is an interpretation function that assigns to every predicate P with arity $Ar(P)$ a set of ordered tuples $P^{\mathcal{I}} \subseteq U^{\mathcal{I}Ar(P)}$. The most obvious definition of the tree structure interpretation is:

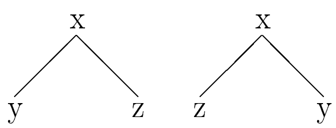
7. (a) $U^{\mathcal{I}} = t_{NL}$
   (b) For all $l \in L$ and $d \in U^{\mathcal{I}}$, $d \in l^{\mathcal{I}}$ iff $(r_d, l) \in L_d$
   (c) For all $d, e \in U^{\mathcal{I}}$, $(d, e) \in Dom^{\mathcal{I}}$ iff e is a subtree of d
   (d) For all $d, e \in U^{\mathcal{I}}$, $(d, e) \in Prec^{\mathcal{I}}$ iff there exists an $e' \in U^{\mathcal{I}}$ such that d and e are subtrees of e' and $(r_d, r_e) \in P_{e'}$

However, a tree interpretation defined in this manner is not suitable as an interpretation for tree constraints. It has a serious drawback in that the precedence relation is no longer an order relation. It will be the case that a pair of trees occur together as subtrees of two different larger trees, but in a different order. For instance, $U^{\mathcal{I}}$ may contain the following four trees:

8. (a) $T_1 = \langle \{x, y, z\}, \{(x, x), (y, y)(z, z), (x, y), (x, z)\}, \{y, z\}, x,$
   $\{(x, A), (y, B), (z, C)\}, \{A, B, C\} \rangle$
   (b) $T_2 = \langle \{x, y, z\}, \{(x, x), (y, y)(z, z), (x, y), (x, z)\}, \{z, y\}, x,$
   $\{(x, A), (y, B), (z, C)\}, \{A, B, C\} \rangle$
   (c) $T_3 = \langle \{y\}, \{(y, y)\}, \emptyset, y, \{(y, B)\}, \{A, B, C\} \rangle$
   (d) $T_4 = \langle \{z\}, \{(z, z)\}, \emptyset, z, \{(z, C)\}, \{A, B, C\} \rangle$

The trees T_3 and T_4 consist of the single root node y and z respectively. The first two trees can be graphically represented as follows:

8. (e) T_1: T_2:



In this case trees T_3 and T_4 are both subtrees of T_1 and T_2, but they occur in a different order. As a result both (T_3, T_4) and (T_4, T_3) will be in $Prec^{\mathcal{I}}$, and $Prec^{\mathcal{I}}$ will not be asymmetric.

There are three alternatives to amend this situation. The first is to abandon tree structure interpretations as possible interpretations for tree constraints. The second is to impose an additional condition on the domain of a tree structure interpretation. The final alternative is to change the definition of the interpretation of the precedence predicate in the definition above. Ignoring the first option, I will discuss the latter two options in turn.

Conditions on $U^{\mathcal{I}}$ The cause of the problems of the precedence relation is that a tree can occur as a subtree in two other trees that do not stand in the dominance relation to each other. This situation can easily be excluded by stating that for all pairs of elements in the domain of an interpretation, their sets of nodes are disjoint or one dominates the other. The definition of a tree structure interpretation then becomes:

9. (a) $U^{\mathcal{I}} = t_{VL}$ such that for every pair $d, e \in U^{\mathcal{I}}$, N_d and N_e are disjoint or d is a subtree of e or e is a subtree of d
   (b) For all $l \in L$ and $d \in U^{\mathcal{I}}$, $d \in l^{\mathcal{I}}$ iff $(r_d, l) \in L_d$
   (c) For all $d, e \in U^{\mathcal{I}}$, $(d, e) \in Dom^{\mathcal{I}}$ iff e is a subtree of d
   (d) For all $d, e \in U^{\mathcal{I}}$, $(d, e) \in Prec^{\mathcal{I}}$ iff there exists an $e' \in U^{\mathcal{I}}$ such that d and e are subtrees of e' and $(r_d, r_e) \in P_{e'}$

This definition retains the properties of the relations of the tree structures.

Ternary Precedence The problems of the precedence relation in a tree structure interpretation can be resolved by taking into account that the precedence order of subtrees is determined relative to a larger tree in which they occur. The precedence relation will then be represented by a ternary predicate. The definition of ($U^{\mathcal{I}}$ and $\cdot^{\mathcal{I}}$) becomes as follows:

10. (a) $U^{\mathcal{I}} = t_{VL}$
    (b) For all $l \in L$ and $d \in U^{\mathcal{I}}$, $d \in l^{\mathcal{I}}$ iff $(r_d, l) \in L_d$
    (c) For all $d, e \in U^{\mathcal{I}}$, $(d, e) \in Dom^{\mathcal{I}}$ iff e is a subtree of d
    (d) For all $d, e, e' \in U^{\mathcal{I}}$, $(d, e, e') \in Prec^{\mathcal{I}}$ iff e and e' are subtrees of d and $(r_e, r_{e'}) \in P_d$

It may be desirable to retain the possibility that a single subtree occurs in more than one larger trees. Consider that in the Chomskyan theories a subtree at one level of representation is often considered to be token-identical to a subtree at another. I will therefore adopt this final definition of tree structure interpretations.

3.3 Tree Constraints

In this section I will present the tree constraints that are interpreted in the domain of tree structure interpretations. These constraints are defined with Predicate Logic in mind, but it must be noted that many alternatives are possible here (for instance, a modal logic of tree structures).

Tree Logic does not have function symbols, and the only terms are variables. Atomic constraints are $p(x_1, \ldots, x_{Ar(p)})$ (where $p \in P$) and $x \doteq y$.

Tree contraints, the formulae of Tree Logic, are defined as follows (ϕ and ψ stand for constraints):

ϕ, ψ	$\longrightarrow$	$x \doteq y$	equation
		$p(x_1, \ldots, x_{Ar(p)})$	predication
		$\phi \wedge \psi$	conjunction
		$\neg\phi$	negation
		$\phi \vee \psi$	disjunction
		$\phi \rightarrow \psi$	implication
		$\exists x(\phi)$	existential quantification
		$\forall x(\phi)$	universal quantification

To interpret the denotation of the tree constraints we require variable assignments, functions that map variables to objects in the domain of an interpretation (ie. functions from Var to $U^{\mathcal{I}}$). For a given interpretation $\mathcal{I}$, an $\mathcal{I}$-assignment α is a mapping from the set of all variables to the domain $U^{\mathcal{I}}$ of the tree interpretation $\mathcal{I}$. Let $\alpha_{x/d}$ be the variable assignment identical to α except for $\alpha_{x/d}(x) = d$ (ie. for all other variables y, $\alpha(y) = \alpha_{x/d}(y)$).

The three-place satisfaction relation $\mathcal{I}, \alpha \models \phi$ holds if the formula ϕ is satisfied by the interpretation $\mathcal{I}$ assuming the variable assignment α. It is defined as follows:

$\mathcal{I}, \alpha \models x \doteq y$	iff	$\alpha(x) = \alpha(y)$
$\mathcal{I}, \alpha \models p(x_1, \ldots, x_{Ar(P)})$	iff	$(\alpha(x_1), \ldots, \alpha(x_{Ar(p)})) \in p^{\mathcal{I}}$
$\mathcal{I}, \alpha \models \phi \wedge \psi$	iff	$\mathcal{I}, \alpha \models \phi$ and $\mathcal{I}, \alpha \models \psi$
$\mathcal{I}, \alpha \models \phi \vee \psi$	iff	$\mathcal{I}, \alpha \models \phi$ or $\mathcal{I}, \alpha \models \psi$
$\mathcal{I}, \alpha \models \neg\phi$	iff	$\mathcal{I}, \alpha \not\models \phi$
$\mathcal{I}, \alpha \models \phi \rightarrow \psi$	iff	$\mathcal{I}, \alpha \models \phi$ implies $\mathcal{I}, \alpha \models \psi$
$\mathcal{I}, \alpha \models \exists x(\phi)$	iff	$\mathcal{I}, \alpha_{x/d} \models \phi$ for some $d \in U^{\mathcal{I}}$
$\mathcal{I}, \alpha \models \forall x(\phi)$	iff	$\mathcal{I}, \alpha_{x/d} \models \phi$ for all $d \in U^{\mathcal{I}}$

A constraint ϕ is *satisfiable* or *consistent* if there exists an interpretation $\mathcal{I}$ and an $\mathcal{I}$-assignment α such that $\mathcal{I}, \alpha \models \phi$. An interpretation $\mathcal{I}$ satisfies a constraint ϕ if for all $\mathcal{I}$-assignments α, $\mathcal{I}, \alpha \models \phi$. A constraint ϕ is valid if for all interpretations $\mathcal{I}$ and all $\mathcal{I}$-assignment α, $\mathcal{I}, \alpha \models \phi$. Two constraints ϕ and ψ are *equivalent* ($\phi \sim \psi$) if for all interpretations $\mathcal{I}$, the set of $\mathcal{I}$-assignments α such that $\mathcal{I}, \alpha \models \phi$ is identical to the set of $\mathcal{I}$-assignments β such that $\mathcal{I}, \beta \models \psi$.

At this moment there has been no investigation regarding the satisfiability of arbitrary tree constraints or of restricted classes of tree constraints. The main interest in developing the formalism was to establish its usefulness as a framework for the expression of linguistic theories in the GB framework.

3.4 Transformations as Relations

Transformations are represented in Tree Logic as relations between trees. The set of predicates is extended with suitable symbols for the general transformation GT and the two instances of Move-α (substitution and adjunction) and a suitable interpretation is assigned to each of these.

The general transformation GT will be represented by the predicate symbol GT. This predicate takes three arguments: the two trees that are combined and the resulting structure. Recall the way in which GT operates: A phrase-marker K_1 is extended with a new root, immediately dominating the original root, and K_2 is inserted as an immediate daughter of this root. In other words, the root of the resulting structure K_3 immediately dominates K_1 and K_2 and no others. The interpretation of GT can then be defined as follows:

11. For all $d, e, f \in U^{\mathcal{I}}$, $(d, e, f) \in GT^{\mathcal{I}}$ iff $d \neq e$, d and e are immediate subtrees of f and for all immediate subtrees g of f either $g = d$ or $g = f$

The two instances of Move-α both leave a trace behind for the moved subtree. Let us first consider replacement of a subtree by a trace. I will assume that the notion "a is a trace for b" is suitably defined and imposes the relevant constraints on the labeling of the root of the trace (the linguistic literature is not specific about the precise nature of traces). The substitution of a trace for a subtree can be defined as follows:

12. T_1 is the result of inserting the trace T_2 for T_3 in tree T_4 iff

    (a) T_2 is a trace for T_3

    (b) T_2 is a subtree of T_1

    (c) T_3 is a subtree of T_4

    (d) $N_{T1} - N_{T2} = N_{T4} - N_{T3}$

    (e) $\forall x, y \in (N_{T1} - N_{T2})\ (x, y) \in Dom_{T1} \leftrightarrow (x, y) \in Dom_{T4}$

    (f) $\forall x, y \in (N_{T1} - N_{T2})\ (x, y) \in Prec_{T1} \leftrightarrow (x, y) \in Prec_{T4}$

    (g) $\forall x \in (N_{T1} - N_{T2})\ (x, y) \in L_{T1} \leftrightarrow (x, y) \in L_{T4}$

    (h) $\forall x \in (N_{T1} - N_{T2})\ (x, r_{T2}) \in Dom_{T1} \leftrightarrow (x, r_{T3}) \in Dom_{T4}$

    (i) $\forall x \in (N_{T1} - N_{T2})\ (x, r_{T2}) \in Prec_{T1} \leftrightarrow (x, r_{T3}) \in Prec_{T4}$

    (j) $\forall x \in (N_{T1} - N_{T2})\ (r_{T2}, x) \in Prec_{T1} \leftrightarrow (r_{T3}, x) \in Prec_{T4}$

These first three of these conditions are straightforward in stating that the inserted subtree must be a trace for the structure that it replaces, that it must occur in the result, and the replaced subtree must be part of the initial tree. The following condition states that the initial tree and the result must share all nodes that do not occur in the trace or the replaced tree. The next three conditions ensure that the dominance, precedence and labeling relations are identical for the nodes that are shared by the initial and the resulting tree. The final three conditions express the fact that the trace and the replaced tree are positioned in the same location with respect to the nodes shared by the initial and resulting tree.

The interpretation of the substitution operation can now be defined. Recall that substitution must always extend the targeted phrase-marker. This means that the resulting phrase-marker must immediately dominate the replaced subtree and the tree that resulted from replacing the subtree by a trace, and no others. In other words, the result of substition is the result of applying GT to the replaced subtree and the tree in which it was replaced.

13. For all $d, e \in U^{\mathcal{I}}$, $(d, e) \in Subst^{\mathcal{I}}$ iff there are $f, g, h \in U^{\mathcal{I}}$ such that f is the result of inserting the trace g for h in d and $(h, f, e) \in GT^{\mathcal{I}}$

To define the interpretation of the adjunction operation we need to make use of an additional notion, namely that of the immediately dominating segment of a node (in a given tree). Segments were briefly described in section 2. A node a is the immediately dominating segment of a node b iff a immediately dominates b and a and b are segments of the same category (all this with respect to a given tree). Adjunction proceeds by inserting an immediately dominating segment for a node and inserting the moved subtree as an immediate daughter of the dominating segment:

14. T_1 is the result of adjoining T_2 to a node n in T_3 iff
    (a) $N_{T1} = N_{T3} \cup N_{T2} \cup \{m\}$
    (b) m is the immediately dominating segment of n in T_1
    (c) m immediately dominates r_{T2} in T_1
    (d) $\forall x, y \in N_{T3}\ (x, y) \in Dom_{T1} \leftrightarrow (x, y) \in Dom_{T3}$
    (e) $\forall x, y \in N_{T3}\ (x, y) \in Prec_{T1} \leftrightarrow (x, y) \in Prec_{T3}$
    (f) $\forall x \in N_{T3}\ (x, y) \in L_{T1} \leftrightarrow (x, y) \in L_{T3}$
    (g) $\forall x, y \in N_{T2}\ (x, y) \in Dom_{T1} \leftrightarrow (x, y) \in Dom_{T2}$
    (h) $\forall x, y \in N_{T2}\ (x, y) \in Prec_{T1} \leftrightarrow (x, y) \in Prec_{T2}$
    (i) $\forall x \in N_{T2}\ (x, y) \in L_{T1} \leftrightarrow (x, y) \in L_{T2}$

The first of these conditions states that the nodes of the resulting tree are the nodes of the original tree plus the nodes of the adjoined subtree plus the added segment. The second and third condition require that the node that immediately dominates the root of the adjoined structure in the resulting tree is the immediately dominating segment of the targeted node. The next three conditions assert that the dominance, precedence and labeling relations that exist for the nodes in the original structure are retained in the resulting structure. The last three conditions state the similar requirement with respect to the tree that is adjoined. Note that there are no constraints on the order between the adjoined subtree and the bottom segment. This allows for both right and left adjunction.

The interpretation of adjunction can now be defined as follows:

15. For all $d, e \in U^{\mathcal{I}}$, $(d, e) \in Adj^{\mathcal{I}}$ iff there are $f, g, h \in U^{\mathcal{I}}$ such that f is the result of inserting the trace g for h in d and e is the result of adjoining h to some node in f.

4 Examples

This section is based mainly on (Haegeman, 1991), which, being an introductory textbook, is necessarily a compilation of a previous version of the theory. The following should therefore be taken as an illustration of the manner in which constraints can be expressed in the formalism rather than as a definitve formalisation.

4.1 Dominance Relations

In addition to the basic dominance predicate *Dom*, the theory employs two other notions of dominance: immediate dominance and reflexive dominance. These can be defined on the basis of the *Dom* predicate:

16. immediate dominance
    $Idom(x, y) \equiv \forall z(Dom(z, y) \rightarrow Dom(z, x) \vee z \doteq x)$

17. reflexive dominance
    $Rdom(x, y) \equiv Dom(x, y) \vee x \doteq y$

4.2 Command Relations

Haegeman Haegeman (1991), p. 135 gives the following definition for generic command relations:

18. A (c-)commands B iff A does not dominate B and every X that dominates A also dominates B.

Haegeman remarks that "when X is equated with the first branching node. A **c-commands** B. When X is interpreted as a maximal projection, A **m-commands** B." It will be obvious that in the case of c-command the notion of *first* branching node is not required. Interpreting X as branching node will have the same result. This definition shows that any property that can hold of nodes gives rise to a command relation (see also (Barker and Pullum, 1990)). The following formula schema shows that any command relation can be expressed if the relevant property (represented by P) can be expressed in the formalism.

19. **Pcommand**
$Pcommand(x, y) \equiv \neg Dom(x, y) \wedge \forall z(P(z) \wedge Dom(z, x) \rightarrow Dom(z, y))$

The properties of being a maximal node (for m-command) or a branching node (for c-command) can be expressed as follows:

20. $Maximal(x) \equiv label(X, max)$

21. $Branching(x) \equiv \exists y, z\ (Idom(x, y) \wedge Idom(x, z) \wedge z \not\equiv y)$

5 Conclusion

In this paper I have presented one possible instantiation of a formal framework for the study of the interaction between transformations and constraints in the Chomskyan tradition. The formalism is based on First Order Predicate Logic and is not geared to one specific instance of Chomskyan linguistics. It must be noted that there are many different instantiations possible and that at this point in time it is hard to determine if one formalism is better than another for the purpose in mind. Alternative formalisms could for instance be based on Modal Logic, or assume a different structure as the representation for Chomsky's phrase-markers. They might possibly adopt a domain-theoretic semantics rather than the standard model-theoretic semantics. The study of the presented formalism and its alternatives will not only contribute to our understanding of the properties of these frameworks, but will also be expected to result in a better insight into the linguistic theories that assume trees (or tree-like structures) as the representation of linguistic objects.

References

C. Barker and Geoffrey Pullum. A theory of command relations. *Linguistics and Philosophy*, 13(1):1–34, 1990.

Patrick Blackburn, Claire Gardent, and Wilfried Meyer-Viol. Talking about trees. In *Proceedings of the 6th Conference of the European Chapter of the Association for Computational Linguistics*, pages 21–29, April 1993.

Noam Chomsky. *A minimalist program for linguistic theory.* MIT Occasional Papers in Linguistics. MIT Linguistics Dept., Cambridge, Mass., 1992.

Liliane Haegeman. *Introduction to Government and Binding Theory.* Blackwell, Oxford, UK., 1991.

Luigi Rizzi. *Relitivized Minimality.* Linguistic Inquiry Monograph 16. MIT Press, Cambridge Mass., 1990.

James Rogers and K. Vijay-Shanker. Reasoning with descriptions of trees. In *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, pages 72–80, 1992.

Gert Smolka. Feature constraint logics for unification grammars. IWBS Report 93, IWBS, IBM Deutschland, 1989. Also appeared in the Journal of Logic Programming 1991.

Herbert Ruessink. Two levels of description: a formal interpretation of transformations. To appear in OTS Jaarboek 1993.

Robert Wall. *Introduction to Mathematical Linguistics.* Prentice-Hall, Inc., Englewood Cliffs, NJ, 1972.