Model Theoretic Syntax

Thomas Cornell
University of Tübingen

James Rogers
University of Central Florida

August 18, 1998

1. Introduction

This chapter is about the principles and parameters (P&P) framework, and it is also about grammar formalisms, formal languages and generative capacity. It is generally believed that these topics parted company sometime in the 1970s at the latest, and now have nothing at all to do with one another. It turns out, however, that the connections between these fields run deeper than formerly suspected. The study of these connections, as they appear in a modern day setting, is the topic of the branch of mathematical linguistics that we are here calling model theoretic syntax.

Following standard P&P assumptions, we take a language, and Universal Grammar more generally, to be appropriately described by a set of universal principles and language particular parameters. A grammar is a theory of a language. If principles take the general form of well-formedness conditions on members of a certain predefined class of abstract structures (phrase markers or tuples of phrase markers), then the most natural formalization of principles is as statements in a logic which are satisfied by all and only the well-formed phrase markers (or tuples of phrase markers). A grammar, then, is an axiomatic presentation of a logical theory, and its relationship to psychologically real objects is an instance of the relationship that holds between any logical theory and its models.

It has been difficult in the past to get much in the way of formal mathematical results out of these logic based grammars, certainly not results of any generality covering entire classes of grammars. Results for particular grammars can be gotten: for example, Stabler (1992) is able to show that the Condition on Extraction Domains (CED) follows from the Barriers theory of Chomsky (1986) even if one removes Subjacency from the theory, but not if one removes the ECP. Chomsky (1986) had related CED effects to Subjacency, and Browning (1989) had argued that the CED should be independent of the ECP, so this is an interesting result, which could not have been obtained without formal logic and automated theorem provers.

However, results for large classes of grammars are much harder to come by. Indeed, the entire notion of "classes of P&P grammars" seems to depend on placing ad hoc and a priori conditions on what can count as a principle or what can count as a parameter. No one so far has been able to propose an explanatory theory of principles or of parameters, nor does such a theory seem likely of discovery. This is in stark contrast to formal grammars in a more classical vein, which retain strong connections to formal language theory. There it has remained relatively straightforward to get worst case parsing complexity results, weak and strong equivalence results between distinct formalisms, decidability results and so forth for quite general classes of grammars (see, e.g., Joshi et al. 1991).

The secret to connecting P&P theories with formal language theory lies in a closer examination of the "formalism" most natural to the P&P framework, namely Logic. While early approaches to formal P&P grammars simply adopted a logic which was guaranteed to be expressive enough for the purpose—Stabler (1992) uses full first order logic, for example—more recent approaches have examined more closely the applicability of much weaker logics to the problem of writing formal P&P grammars. These weaker logics turned out to be connected to formal language theory either via the theory of context free grammars or the theory of tree automata, yielding definite expressibility results. These expressibility results take the general form: a set of trees is definable in logic L if and only if it is also definable using formal language theoretic means X.

It may be important to keep in mind that the tenor of this research is analytic. There is no a priori assumption that grammars should be restricted in their language theoretic complexity. The question instead is, essentially, "Given theory T, what is the weakest logic I can use to express T?" Or, more generally, "How much of T can I express in logic L?" These are already interesting questions, since their answers shed light on the structure of a given theory, for example by showing which principles become inexpressible when we weaken the logic a notch. However the connection between Logic and Computation—the strength and richness of which has only recently begun to be appreciated—allows us also to hope, as in the old days, that uncovering structural aspects of human languages will also shed light on the nature of the computational system of human language qua computational system.

1.1. Logical approaches to syntax

Model theoretic syntax is not to be construed as simply using logic to write grammars. The formalization of linguistic theories in logic is a much broader field. Historically, we can trace its roots to computational linguistics, where the need for increased precision in specifying grammars together with the rise of deductive approaches to parsing led naturally to the implementation of grammars as logic programs. Much of the early work in using logic to write grammars was thus motivated by the needs of the computational linguistics community.

The needs of computation and the needs of mathematics actually impose rather different constraints on logic, with the result that two distinct strains have become visible among logical approaches to grammar. Mark Johnson (1994) has referred to these as "validity based" and "satisfiability based". For our purposes, the distinction is reflected very directly in the kinds of models that are assumed. In validity based approaches the well-formed structures become individuals in a model over various kinds of objects, including trees and possibly also lists, tuples and other data types, together with features or feature structures; over this universe are defined various predicates and relations. So, for example, x being a constituent of y is modeled via a relation that holds between two distinct individuals. The advantage of this approach is that one can talk about the well-formedness of a particular structure directly, since the whole structure has an encoding in the logical language. This makes it very useful for computational work, where, for example, one would want a representation of the structure of an input sentence as the output value of a parser, and indeed all work that we know of in the principle based parsing as deduction field follows this line (e.g., Stabler 1992; Johnson 1991; Berwick et al. 1991; Macías 1990; Fong 1992). The drawback of this approach is that the set of models no longer represents any particular linguistic concept. For example, the language itself is a strict subset of the domain of discourse. Also, the models of such a grammar can change drastically when the grammar is rewritten, since even auxiliary relations and predicates must be interpreted in the model. So this approach is less well suited to mathematical work, compared to the satisfiability based approach.

One might recast Johnson's distinctions as "language based" and "model based". In validity based approaches to grammar the emphasis is on the language, being able to express directly the data structures of interest and the relations that hold among them. In satisfiability based approaches, or at least in those that fall under the heading of model theoretic syntax, it is the structures, the models that drive the development of the system.

The approach of model theoretic syntax is to pick out individual well-formed structural descriptions as models for a given set of axioms. Since the structures are meant to be linguistic entities, the description language appropriate to them is accordingly restricted. As a result model theoretic syntax tends to employ languages with a very small number of primitive relations, as opposed to logic programming oriented approaches, which, because of their reliance on recursive "definitions", must employ a very large and rather amorphous set of primitives. Still, the distinction should not be overemphasized. One might be able, for example, to recast such theories using, in place of first order logic, first order logic with fixed point operators (or a comparable second order logic). This strictly increases the power of the logic by allowing (some) recursive definitions. As far as we know no one has pursued this in linguistics, but it seems likely that it would significantly lessen the difference between these two approaches.

1.2. Formal languages and "E-Languages"

The standard position in P&P theory on the relationship between human languages and formal languages has typically been presented as a contrast between I-language (internal, intensional language) and E-language (external, extensional language). Unfortunately, every work we know of that treats this subject identifies E-languages with sets of strings. We need much more than that to state clearly what formal language theory is about, however. In particular, we will need some way to talk about sets of trees as formal languages, either as the derivation trees (strong interpretation) of a string grammar or as the output (weak interpretation) of a tree grammar.

It is puzzling that only sets of strings get to be E-languages. It seems reasonably clear that sets of trees, or any other structure of choice, are equally extensional and external. Minimally, we need to distinguish a notion of external structural description (SD) from the (internal) structural descriptions of ordinary linguistic usage. In P&P theory, an SD is an aspect of human psychology (and consequently not accessible to direct inspection). In the following, we will define a formal SD as a mathematical construct, defined in the usual stipulative way and intended to capture working hypotheses about "internal SDs". Given this notion of "external SD", we can distinguish both strong and weak E-languages—sets of (formal) structural descriptions as well as sets of strings—from I-language proper, that is, the actual generative procedure constituting the core of the human language faculty.

Accordingly, a formal description of an I-language—a formal grammar, that is—generates both strong and weak E-languages. Its strong E-language will just be the set of all formal structural descriptions which are licensed by the grammar. Chomsky (1995) refers to the **structure of an I-language**, meaning the set of SDs licensed by the I-language. The strong E-language generated by a grammar is thus an externalization of the structure of the I-language of which the grammar is a theory. The weak E-language will then be the **yield** of the strong E-language, that is, the strings of (phonetically realizable) terminal symbols yielded by each SD. The day to day work of syntactic theory can be seen in these terms as the attempt to capture the structure of an I-language (in Chomsky's sense) using (strong) E-languages.

This apparatus, of strong and weak E-languages over externalized structural descriptions, is useful in discussing connections between transformational grammar and phrase structure grammars of various kinds. However, it still misses the point slightly. The phrase structure grammars which give rise to these notions of strong and weak generative capacity are in some sense "really" string grammars: their strong interpretation is a set of **derivation-structures**, which say how the string was derived. The usefulness of phrase structure grammars has always rested on the fact that the derivation structures of a string generating grammar provide an immediate constituent analysis of the string, which is very useful and interesting when one is focused on immediate constituent analysis. But transformational grammars are not in this sense string grammars. They are really designed to yield trees, and to present analyses of sentence structure that go far beyond immediate constituent analysis. So, in this important sense, even if we focus on the strong interpretation of a string grammar, it is still the case that the grammar is really a string grammar while the transformational grammar is really a tree grammar. Seen as formal grammars, transformational grammars are meant to externalize the structure of an I-language much more directly than the externalization via string grammars.

There is, however, also a formal notion of tree grammar, well studied within formal language theory (Gécseg & Steinby 1997, and refs.). Within the theory of formal languages virtually everything that has been done with strings has been generalized to trees. For example, we can distinguish regular and context-free **tree grammars** just as we have regular and context-free string grammars (Maibaum 1974; Engelfriet & Schmidt 1977, 1978; Mönnich 1997a). Furthermore, tree grammars and string grammars are related in tight and interesting ways. For example, the regular (usually called "recognizable") tree languages yield the context-free string languages, and the context-free tree languages yield the indexed (string) languages of Aho (1968). (As far as we know "indexed tree grammars" have not been investigated.) Similar generalizations have been made within the study of automata, of algebraic characterizations of languages, and, most importantly for our topic here, within the study of logics that characterize languages (Thatcher & Wright 1968; Doner 1970; Rabin 1969). Indeed, it now appears that the same generalizations can be pushed even farther, from trees to structures of even higher dimension (Rogers 1997b, 1998).

To be sure, things are more complex when we deal with trees, and more design options present themselves. For example, while the recognizable (i.e., regular) tree languages yield the context free string languages, the derivation structures (strong languages) generated by context free string grammars form a strict subset of the recognizable tree languages. Whether this difference is significant is debatable—we will take it up below—but it serves as an example of the subtleties one encounters when one makes trees the primary objects of formal grammars. As another example, we know that leftmost and rightmost derivations from a string grammar yield the same sets of strings, but "uppermost" and "lowermost" (called outside-in and inside-out) derivations from tree rewrite grammars do not yield the same sets of trees (Fischer 1968; Engelfriet & Schmidt 1977, 1978). Also, if "nonterminals" of the grammar are restricted so that they connect only one lower tree to its host upper tree then we get (to a first approximation) the class of tree languages studied in the framework of Tree Adjoining Grammar (Mönnich 1997a). More generally, new rungs appear in the Chomsky hierarchy between the context free and context sensitive (string) languages. The landscape is richer when we look at trees, but everything that formal language theory has done with strings appears as a special case in this setting: the case of a nonbranching tree turned on its side, essentially.

The attempt to characterize the mathematical study of formal languages as the study of sets of strings, then, is a grave misunderstanding of what the theory is about. To be sure, it is a misunderstanding that the theoretical computer science literature has done as much to foster as the theoretical linguistics literature. For example, standard introductory texts such as Hopcroft & Ullman (1979) typically do not mention tree grammars or tree automata at all, even though the original generalizations of string languages to tree languages date back to the mid-60s. This is all the more puzzling since formal languages as they are actually used, on the other hand, are really linear representations of trees. For example, an expression like $x + (5 * y)$ is as much a "string" as the labelled bracketing $[_+ \; x \; [_* \; 5 \; y]]$. Both are, in fact, and obviously so from their interpretations, linear representations of trees.

Given this recognition of strings not only as the yield of trees but as a special case of trees themselves, our terminology of strong and weak languages must be used with care. In particular, a set of trees may be defined either as the strong language of a string grammar or the weak language of a tree grammar. A string language may be defined as the yield of a tree language without regard for whether that tree language is generated as the strong language of a string grammar or the weak language of a tree grammar. It is important not to lose sight of the close connections between these two ways of defining string languages, however. Since the context free string languages can be got either as the yield of the trees strongly generated by a context free string grammar or as the yield of the trees weakly generated by a regular tree grammar, "weak generative capacity results" for natural languages still have an impact on the theory of tree grammars for natural languages. However, there are also differences: as noted, the sets of trees generable from a context free string grammar are a strict subset of the trees generable from a regular tree grammar. How significant this difference is will be disussed below.

1.3. Are structural descriptions trees?

We will also assume throughout that structural descriptions are reducible to phrase markers, and that phrase markers are trees. These are not innocent assumptions! First off, by reducing SDs to phrase markers we are effectively ignoring multi-level approaches, like classical GB theory where the models would be 4-tuples of structures $(\mathrm{DS}, \mathrm{SS}, \mathrm{LF}, \mathrm{PF})$, together with a relation Move-α or Affect-α, or more purely derivational approaches to grammar, such as categorial grammar (Moortgat 1996; Morrill 1994) and most work in the minimalist program (Chomsky 1995). By restricting our attention to trees, we are also ruling out a considerable body of linguistic research. Outside of transformational grammar we have seen the rise of (typed) feature structures (Carpenter 1992; Keller 1993, and refs.) as linguistic representations, and even within transformational grammar there has been a steady stream of proposals from the very earliest days which offer types of phrase markers which are not (quite) trees. For example, the original P-markers of Chomsky (1975/1955) were not exactly trees. Chametzky (1996) offers a good summary of this literature.

To a considerable extent the techniques we are interested in here are applicable to these other kinds of structures as well. Indeed the original impetus for model theoretic syntax stems fairly directly from work in feature structures: feature structures began their lives as formulas with no clear formal interpretation and work on the semantics of feature structures brought model theoretic techniques into linguistics at an unprecedented level. Any linguistic structure, be it customized phrase markers or even T-markers representing entire transformational derivations, can be associated with a description language, and hence a formal theory. Whether they would have as nice mathematical properties as trees is hard to tell in advance of an in depth study. (Though see Cornell (1997a,b); Kolb (1997); Gärtner (1997) and Lecomte (1997) for steps towards at least a theory of transformational derivation structures.)

2. Describing trees

In order to talk about grammars or classes of grammars we first need some way of talking about trees. The formal language in which a logic (in our case, a tree logic) is presented can be split into two pieces: the logical constants and the parameters. In the general study of logics (e.g., classical first order logic, monadic second order logic, modal logic, propositional dynamic logic) one usually decides on a set of logical constants, whose interpretations are fixed, that is, they are built into the very definition of "interpretation". For example, in first order logic with equality one might choose as constants the existential quantifier over individuals, conjunction, negation and equality. Then, for a particular application of this logic, one decides on a set of parameters (predicate symbols, relation symbols, individual constant symbols) whose interpretations may differ freely from model to model. A model is, in fact, defined as a particular interpretation of a given set of parameters. Models differ inasmuch as they give a different interpretation to the parameters. If one is studying group theory, for example, then one's models (groups) will always have to offer some interpretation of the multiplication operation, but different groups will offer different interpretations. And if one is studying first order logic in general, then one will allow the sets of parameters to vary freely, keeping only the logical constants fixed.

In studying the collection of tree logics employed in model theoretic syntax we will do almost the opposite: the set of parameters will remain relatively fixed and we will vary the logical apparatus with which we put them into play. This reflects our concern with a particular sort of structure. There are only a few primitive relations that make sense when one is talking about trees: dominance, precedence and labeling of individual nodes, chiefly. As we vary the logical apparatus (quantifiers over individuals, quantifiers over sets, fixed point operators, modal operators, etc.) we will be able to describe different sets of models, that is, different sets of trees. But at the core of the research program lies the necessity of being able to offer some description for any given single tree.

There are two main branches of model theoretic syntax, which lead to tree description languages that are superficially rather different, and which lead to rather different habits of thought about tree structures. One branch uses classical predicate logics (Rogers 1997a, forthcoming, 1994; Palm 1997; Cornell 1994, 1992) while the other uses modal logics (Kracht 1995b, 1993a; Blackburn & Meyer-Viol 1994; Blackburn et al. 1995; Palm 1997). In almost all cases the underlying relations are the same: left sister, immediate dominance, first, second, $\ldots$, k-th daughter, and the inverses, compositions, and reflexive, transitive closures of these. Nonetheless, they appear rather differently depending on whether the surrounding apparatus is a predicate logic or a modal logic.

This is not the place to give a thoroughgoing introduction to either sort of logic, but perhaps the basic flavor can be got across in brief. Let us fix for the moment the following symbols to represent a few of the structural relations that are crucial to describing trees. Let $\mathbf{d}_i$ denote the relation that holds between a node and its i-th daughter, $\mathbf{d}$ denote the relation that holds between a node and each one of its daughters, $\mathbf{u}$ the relation that holds between a node and its parent, $\mathbf{l}$ and $\mathbf{r}$ the relations that hold between a node and its left and right sister, respectively. As a general notational device we will attach a $*$ to a relation symbol to denote its reflexive transitive closure and a $+$ to denote its irreflexive transitive closure. So, for example, the remote dominance relation, i.e., the relation that holds for example between the root and every node of a tree, will here be symbolized $\mathbf{d}^*$. The strict dominance relation, i.e., the relation that holds for example between the root and every node in the tree other than itself would be symbolized $\mathbf{d}^+$. We will not fix a collection of symbols for node labels. We will assume that any alphanumeric string will do. Also we will allow nodes to be multiply labeled, so that we think of labels as features of the node: given label L, a node is $[+L]$ if it is labeled L and $[-L]$ if not.

Relations and labels are used in different ways in predicate and modal logics. In predicate logic we suppose we have a collection of variables ranging over nodes in a tree. Then, supposing we have two variables x and y, we write $x\mathbf{d}^*y$ or $\mathbf{d}^*(x,y)$ to mean that the node that x stands for dominates the node that y stands for. We use quantifiers to say things like "whatever x stands for" or "something that x might stand for". For example, $\exists x \forall y\ \mathbf{d}^*(x,y)$ is a formula of first order logic satisfied in a tree by assigning x to the root. The node labels will typically be used as one place ("monadic") predicate symbols: $N(x)$ should be interpreted for example as a claim that the node that x denotes is labeled N, i.e., has the feature [+N].

In (propositional) modal logic the role of the quantifiers is taken over by the modal operators. Variables are not used; instead the feature-labels appear as propositions. In general, we assume for each relation of interest that our description language contains two modal operators for that relation. Where there is only one underlying relation we would write $\Diamond$ and $\Box$ for the two operators that (roughly) play the role of existential and universal quantification in a predicate logic. Here we will write $\langle\mathbf{d}^*\rangle$ and $[\mathbf{d}^*]$, for example, and interpret these complex symbols to mean "somewhere below" and "everywhere below", respectively. Node labels in a modal tree logic will typically appear as proposition symbols. The models are trees of course, and the various different relations are given their usual interpretations in trees.

Modal formulas are evaluated at particular nodes of the tree. The simplest formula would just be a single label, and it would be true at a given node just in case that node indeed bore that label. Then the modal operators provide a way of getting around in a tree. For example a formula like $\langle\mathbf{d}_1\rangle N$ should be interpreted as being true at a node just in case, if you visit that node's first daughter, you will find that it is labeled [+N]. Note that the formula is not evaluated at the target node (the [+N] daughter): it is true or false of the starting point. As such it expresses the property of having a [+N] first daugher, not the property of being a [+N] first daughter. In this way modal formulas can express the well-formedness conditions that hold on a node in virtue of the presence of other nodes that are required to license it, for example. Complex modal prefixes have the effect of constructing more elaborate relations from primitive ones. For example, the sequence of modal operators $\langle\mathbf{u}^*\rangle\langle\mathbf{r}^+\rangle\langle\mathbf{d}^*\rangle$ "implements" the (remote) precedence relation, in the sense that a formula

$\langle \mathbf{u}^* \rangle \langle \mathbf{r}^+ \rangle \langle \mathbf{d}^* \rangle \mathsf{F}$ will be true at a node s just in case s precedes some other node with the feature F. A modal formula which is true at all points of a tree is said to be valid in that tree. In this way modal formulas can be used to pick out sets of trees (tree languages), namely those for which that formula is valid.

2.1. Hierarchies of logics

Perhaps the most powerful limitation on the expressiveness of a logic is the general restriction that formulas must be finite. Most of the sophisticated devices built into logical machinery exist to express—finitely—things that would require infinite formulas in a weaker logic. For example, the first order universal quantifier is a finite way of expressing what could be, in propositional logic, an infinite conjunction: "all men are mortal," being roughly equivalent to "man #1 is mortal and man #2 is mortal and...." In the context of tree logics this restriction shows up upon the following observation, that the collection of relations given above may be rather redundant. For example, intuitively it should be possiblt to define $\mathbf{d}^*$ in terms of $\mathbf{d}$. That cannot be done in first order logic or in propositional modal logic, but it can be done in monadic second order logic, or indeed in the weaker first order logic with transitive closure operators, and it can be done in propositional dynamic logic, a form of modal logic.

In predicate logic, extra expressive power beyond what first order quantifiers can do is supplied by devices such as transitive closure operators, which allow one to refer to define the transitive closure of any first order definable relation, least fixed point operators, which are a generalization of transitive closure, and second order quantifiers, usually restricted to monadic second order variables, which allow one to quantify over sets of nodes (e.g., trees) as well as over the individual nodes themselves.

In modal logic extra power comes from the use of propositional dynamic logic. This allows one to define new modal operators from old using the constructors union ("or"), composition ("and then"), the Kleene star ("as often as you like") and test ("if ϕ holds"). This means that, from a small set of primitive relations like "go up", "go down", "go left" and "go right" modal operators can be constructed which are essentially computer programs over these built in operations. As a simple example, consider a formula like $\langle \mathbf{u}; (\mathbf{l} \cup \mathbf{r}); (\mathbf{d})^* \rangle \phi$. This formula holds of a node if it c-commands some other node with property ϕ (abstracting away from problems introduced by nonbranching nodes and the distinction between categories and segements of categories). That is, we can define a c-command modal operator directly in propositional dynamic logic. Note that this simple example could also have been executed in ordinary propositional modal logic, but only if $\mathbf{d}^*$ is admitted as a primitive relation: it cannot be defined in terms of $\mathbf{d}$ in that weaker logic.

The various tree logics built up around this given primitive stock of relations form a partially ordered hierarchy, then. This can also be seen in the sets of trees that are definable in them, as one would expect. For example, to take some artificial examples, the set of trees labeled with a apart from one node labeled b is recognizable. That is, it can be generated by a regular tree grammar and defined in monadic second order logic or propositional dynamic logic, for example. However, there is no context free string grammar which has such trees as its derivation trees, nor can it be defined by a first order tree logic, for example. Nonetheless, the yield of this set of trees is still weakly context free; that is, there are weakly equivalent context free string grammars, just no strongly equivalent ones. So there is evidently some room between the tree languages strongly generated by context free string grammars and the tree languages generated by regular tree grammars, despite their weak equivalence, and the various sorts of tree logics are spread out across this intermediate landscape. (See Volger (1997) for a compact review of a number of the tree logics discussed here and their relationships.)

The question then arises how much difference this different expressive power makes, and whether there could be linguistically significant properties that depend on the logical apparatus surrounding the basic core of relations. The answer is probably not, though the reader may have to formulate their own opinion, for the following reason. Consider again the "one b" example. The reason that this language is generable by a regular tree grammar has to do with this kind of grammar's ability to capture more context then is possible with string rewrite rules. However, this very limited context sensitivity is not really beyond the power of a context free string grammar: if one adds a single feature to the language which is assigned to constituents containing a b somewhere, then b just has to expand into a string of a categories all marked "no b's down here". This is essentially the same kind of trick that appears in formal grammar in the "slash" categories of GPSG, which are used to pass nonlocal information about gaps and fillers around the phrase structure trees generated by (a grammar equivalent to) a context free string grammar. So, if one constructs a grammar with extra features, and is then allowed to ignore them, to "project away from them", then one can get even with a context free string grammar all of the trees that one can define with monadic second order tree logic. However, there is a real gain in succinctness, as this example also shows, in the more powerful logics. This succinctness and expressiveness is important in doing formal work with grammar: it makes the outlines of the original theory appear much more clearly in the formalization.

3. Tree logics in linguistics

3.1. Early history of tree logics in linguistics

Trees enter linguistics as suitable representations for immediate constituent analyses of sentences. One of the fundamental observations in early generative grammar was that the derivations of strings from certain kinds of rewrite systems could be made to yield trees, yielding thereby immediate constituent analyses of the sentences they were deriving. This opened the door to a large body of mathematical tools that were then being developed in the study of the foundations of mathematics and computation, and offered the hope that the technical problems of deriving infinite variety from finite means could at least be seriously addressed, if not solved.

Probably the first step towards a truly model theoretic perspective on tree structures was taken in McCawley's paper on the Base (McCawley 1968), where he argued that rewrite rules ought to be reinterpreted as **node admissability conditions**. Trees—the ones we're interested in at least—arise as members of that set of general mathematical structures which happen to satisfy a given set of node admissability conditions. One can easily see in this the ancestor of the approach which came to fruition in Chomsky (1981), of allowing modules of the grammar to overgenerate, but pruning away all structures which were not tolerated by all the modules together. So let us make bold to rephrase this observation of McCawley's in more modern terms, choosing to look at phrase structure rules as either rules (showing how to construct more complex trees from simpler ones) or principles. For context free grammars it can be proved that the point of view we take does not matter: we get the same languages at the end of the day. However, McCawley also observes that for context sensitive grammars, the view we take is critical. Here, in a certain sense, rules and principles diverge for the first time: interpreted as node-admissibility conditions, context-sensitive grammars do not yield all of the context-sensitive languages. Peters & Ritchie (1973) later showed that context-sensitive grammars so interpreted yield only the context free languages, in fact.

This constraint-based, 'logic-oriented' point of view gained strength through the 1970s. The view of phrase structure grammars as sets of node admissibility conditions was the cornerstone of GPSG. Within transformational grammar, since the early 1960s more and more generalizations about classes of transformational rules were being expressed as constraints on possible transformations. With Chomsky (1981), the theory of transformations was hypothesized to be characterizable (almost) purely in terms of constraints on representations. The content of transformations themselves was reduced to "move anything anywhere" (Move α), or perhaps even "do something to anything" (Affect α). The real work was being done by conditions that the transformations had to satisfy, that is, the transformational relations between levels of representation were defined axiomatically, like the properties that defined the individual levels themselves.

3.2. Recent history of tree logics

Among other things, the 1980s were marked by the rise of a new approach to syntactic representations, namely as **feature structures** (or, more accurately, "attribute value matrices"). Trees, one would think, are pretty mundane, utterly familiar sorts of structure, and if one is only interested in their mathematical properties one needn't do much more than consider the axioms in McCawley (1968), or in Wall (1972), or in Rodman (1977), all of which say pretty much the same thing. Trees we thought we knew—though the reader might be rewarded in comparing the works just cited with more recent work, such as Backofen et al. (1995) or Blackburn et al. (1995). But feature structures really required some serious investigation. Feature structures began as formulas; it was reasonably obvious what they were intended to mean, but the question arose: Did they really mean that? As the notation became richer the question became more interesting. As a result, the 1980s saw the appearance of a string of papers presenting various logics for feature structures and for the grammatical formalisms that were based on them, with the effect that a very large battery of techniques from mathematics—and model theory in particular—became available to the computational and mathematical linguistics community, and, as usual, skills were sharpened by large-scale debate on clearly defined issues. Perhaps more than anything else, the standards for grammar formalisms increased, witnessed mainly by a heightened awareness of and interest in the semantics of formalisms, with its attendant issues of completeness and decidability, as well as issues of equivalence and other relations among distinct formalisms.

Interest in trees as structures interpreting some formal description language lagged behind interest in feature structures with one main exception. In 1983, Marcus, Hindle and Fleck presented a short paper suggesting the advantages of an approach to trees focusing on (possibly remote) dominance rather than on immediate dominance. Since then a number of papers have appeared suggesting the usefulness of such an approach (Vijay-Shanker 1992; Berwick & Weinberg 1985; Weinberg 1988; Marcus & Hindle 1990) (and more recently Gorrell 1995; Weinberg 1994; Sturt & Crocker 1996), but at first the effort of the field was all in motivation and nothing was said of formalization; the ideas were there, but their consequences were not worked out.

Modern work in tree logics, and, more generally, what has come to be known as Model Theoretic Syntax, really begins with two main lines of investigation. On the one hand there was (finally) an attempt to formalize and extend the "D-theory" of Marcus et al. (1983) using the kinds of logical tools made available by studies in feature logics (Cornell 1992; Rogers & Vijay-Shanker 1992). On the other hand, beginning with a seminal paper by Barker & Pullum (1990), there was an interest in the logical underpinnings of command relations (Kracht 1992, 1993a; Blackburn et al. 1993; Blackburn & Meyer-Viol 1994).

The study of command relations begins from the observation that, when linguists look at trees, certain sorts of structural relations become very important, namely the various sorts of command relations that underly key concepts like government and binding. These relations generalize the primitive relation of scope-taking that can be seen in mathematics wherever variable binding operators appear. For example, the quantifier in a formula $\forall x \phi$ binds an ocurrence of x just in case it c-commands it. In linguistics, this elementary concept has been elaborated to other kinds of structural relations between an element and its scope (m-command, "true" c-command—where nonbranching nodes are transparent to the relation—and government, where there is a lower limit on how far an elements binding power can reach).

It turns out, unfortunately, that the mathematical structure of command relations is too weak to get much in the way of results. For example, though asymmetric c-command can be turned into a (strict) partial order, standard c-command has very few mathematical properties. The attempt to improve the situation has almost always led to the abandonment of the "anti-domination condition" (α c-commands β iff ... and α does not dominate β). The results are then hard to use within linguistics, where this condition is almost invariably assumed. Nonetheless, if one makes certain very strong assumptions about the kinds of trees one is allowed to consider as models, then more interesting properties fall out. (Kayne's recent work (Kayne 1994) has been the most extensive investigation of what these conditions might be.) But of more lasting use to the field of tree logics and model theoretic syntax was the natural usefulness of modal logical languages for the description of these sorts of relations.

Interestingly, one recent line of research (Frank & Vijay-Shanker 1998, 1995) has combined these two historical lines of thought. Frank and Vijay-Shanker observe that, where remote dominance appears in descriptions of grammatical structures and principles, it invariably appears there because of a c-command constraint. Their hypothesis is, then, that c-command is actually the relevant primitive, and dominance constraints between nodes are just a reflex of the nature of c-command. From this point of view, these two strands of formal research are bound together around the central concept of the mathematics of remoteness in grammar.

4. Principle based grammars for context free languages

The connection between logics and formal languages is based on two properties that some classes of formal languages have. First are **closure properties**. For example, the regular (string and tree) languages are closed under operations such as intersection or Kleene star (iteration), that is, for example, the intersection of two regular string or tree languages is still regular. The second important properties are **decidable properties**, the most important of which is the decidability of the emptiness of a language. For example, it is not always obvious, indeed it is not always decidable, by inspection of a generative device (a rewriting grammar or a finite automaton), whether it generates or accepts any structures at all.

The connection between languages and generative devices is generally established by an inductive argument. (For a detailed introduction to the technique, see Thomas (1997); Morawietz & Cornell (1997a).) In this sense there is a strong similarity between Rogers' and Kracht's work, even though Rogers is working with monadic second order (predicate) logic and connecting it to tree automata, while Kracht is working with propositional dynamic (modal) logic and connecting it to context free string grammars. The induction begins with the base cases, which involve the proof that the atomic formulas of the logic correspond to appropriate generative devices. In the case of monadic second order logic this means presenting an automaton that recognizes satisfying instances of formulas like $\mathbf{d}^*(x, y)$. In Kracht's case, the elementary building blocks are slightly more complex: they are formulas which regulate the distribution of features according to a constraint, so the role of atomic formulas is played by a biconditional of the form $\mathsf{X} \leftrightarrow \langle R \rangle \mathsf{F}$, which states that the appearance of feature X is licensed just at nodes where $\langle R \rangle \mathsf{F}$ holds, where it is really the $\langle R \rangle$ that are playing the role of atomic formulas. So, for example, one has to exhibit a grammar that will properly distribute X according to the condition $\mathsf{X} \leftrightarrow \langle \mathbf{d} \rangle \mathsf{F}$. This is just a grammar that accepts all trees (for practical purposes we can consider only binary and unary branching trees) in which X appears (or equivalently $\langle \mathbf{d} \rangle \mathsf{F}$ appears) just in case one of its daughters is labeled F. Clearly, this is not hard to do. (Details are presented in Kracht (1995b, 1993b).)

The inductive step is the one that depends on the closure properties of the generating devices in question. Larger formulas are built up from smaller ones by connecting them together with logical constants (like conjunction) or by applying logical operators (like quantifiers or modals) to them. So the inductive step demonstrating the correspondence between a logic and a class of language defining devices must show a correspondence between logical constants (ways of building up big formulas) and operations on the devices (ways of building up more complex devices). Then, if that class of devices is closed under that operation, it is clear that the correspondence is maintained under the construction of larger formulas from smaller. For example, it is possible to construct an automaton that recognizes the intersection of two languages from the automata recognizing those languages. If each "smaller" automaton recognizes the satisfying instances of a formula, then the intersection-automaton recognizes those cases which satisfy both formulas, that is, the cases that satisfy the conjunction of those formulas. Things are more complicated for the case of Kracht's logic, but the pattern is the same: for every way in which new formulas can be built up from old, on the assumption that the old formulas correspond to grammars, a construction on grammars is exhibited that yields a result that corresponds to the new formula.

One sees immediately a corollary to these correspondences. Since formulas from Rogers' monadic second order tree logic correspond to tree automata and formulas from Kracht's propositional dynamic tree logic correspond to context-free grammars, and since both devices can define only context-free string languages, it follows that any set of formal structural descriptions that can be defined by such a grammar must be weakly context free. This is in a certain sense a mathematical accident: tree automata and context free grammars have good, workable mathematical properties, and they therefore represent a good starting point for investigations into the connection of logics to formal grammars and automata. However, these logics were not developed with such results in mind; they were developed with an eye towards presenting a logical vocabulary that would be convenient for the expression of the kinds of properties that linguists observe of natural languages. In particular, Rogers' work arises from the tradition starting from Marcus et al. (1983) and Kracht's from the tradition going back to Barker & Pullum (1990), both traditions being firmly rooted in the concerns of linguistics.

Indeed, it is not easy to imagine linguistically interesting structural relations in trees trees that cannot be described in such powerful languages. As an illustration, Rogers presents a detailed formalization of a large fragment of a GB grammar for English in his logic. Rogers' fragment presents a thorough account of all types of syntactic movement (including rightward movement)—cast in the Relativized Minimality framework of Rizzi (1990)—as well as binding and control. It seems quite plausible now that there is no construction of English which requires more descriptive power than that provided by monadic second order tree logics.

Whatever one may think of the inherent interest of questions of language theoretic complexity, these are remarkable results. Certainly no one casually viewing P&P theories as they developed through the 1980s would have suspected that there was any way of getting such a result: the structure of the underlying framework simply appeared too loose. Anybody gambling on the expressive power of GB theory would almost certainly have bet on a much higher rung of the Chomsky Hierarchy. However, certainly as interesting as the question of what can be said in monadic second order logic or propositional dynamic logic is the question of what cannot be said, and there are indeed aspects of natural language and P&P theories which do fall outside the expressive power of these logics. There are two main directions in which these formal approaches are interestingly limited. First, the claim that an adequate theory of Universal Grammar and its parameters can be written in such a logic appears to be falsified by certain data from languages other than English. In particular, there are languages like Swiss German which are argued (Huybregts 1984; Shieber 1985) not to be even weakly context-free, and certainly the analysis of Dutch or even German verb-clusters that is standard in P&P analyses does not define a recognizable set of trees, essentially by the same arguments. This is an example where otherwise rather uninteresting weak expressiveness results can be useful: they indicate a lower bound in the expressive power of grammars for the given language. In this case, they indicate that whatever mechanism could adequately constrain complex head movement phenomena must be inexpressible in Rogers' monadic second order tree logic and so, a fortiori, in any of the other tree logics we have mentioned.

The second major limitation is more formal and theory internal, and depends less on empirical results. Rogers proves that the ability to express the mechanism of Free Indexation lies strictly outside the means of monadic second order tree logics as we have considered them so far. These two inexpressibility results—for germanic head movement constructions and for free indexation—may be related. Naively, the problem with free indexation is that it presupposes an infinite number of distinct properties that nodes can have, since each index is distinct from all the others and there are arbitrarily many of them. Naively, that would require an infinite second order formula even to define which node properties were indices, or what it would mean to have the same index. Of course, if one could quantify over sets of sets then one could use a quantifier in place of such an infinite formula, but that takes one into third order logic, and even monadic third order logic is too powerful. In fact, Rogers' result is more subtle than this analysis indicates: he proves that any device which is equivalent to free indexing will drive the logic into undecidability and so, a fortiori, out of the range of monadic second order tree logics. This entails several other results as well: this logic lacks the power to define equality or even weaker isomorphisms between distinct subtrees. Therefore it would be impossible to implement the copy theory of movement, for example, since it would be impossible to require one subtree to be a copy of another.

Having to do without indices, Rogers' formalization must make do with grammatically identifiable, locally unique properties of chain links. Here the framework of Rizzi's Relativized Minimality comes in handy: for example the nearest c-commanding $\bar{A}$ specifier must be the antecedent of a given variable, irrespective of any indexing. In effect, chain types take the place of indices. Obviously, as a result chains of the same type may not overlap: there would be no way to tease them apart, as they have, effectively, the same index. Apart from Rizzi's system, other linguists have proposed theories with similar effects. For example it follows from the proposals in Manzini (1992) that chains of the same type cannot overlap, and indeed this is also a consequence of the Minimal Link Condition of Chomsky (1995), though in a derivational setting, where, under certain circumstances, potential intervening elements may be deleted from the representation, it is a little harder to gauge the consequences of a minimality-based approach to locality. (It is certainly clear (Stabler 1997; Cornell 1996) that formalizations of this approach can define artificial languages that are known to lie outside the boundaries of context freeness.) Finally, this limit on simultaneously active chain links fits in well with approaches to the limits of human language processing resources as outlined for example in Stabler (1994) or Gibson (1991, 1995).

This limitation on overlapping chains is precisely the problem posed by germanic head movement constructions. Since the hierarchical order within the complex head reflects the hierarchical order of constituents which depend on that head (for example its arguments), the chain links connecting the head to its trace position amongst its arguments must overlap. Indeed, this pattern of crossing dependencies is characteristic of noncontext free languages.

It is interesting, then, that so little of Universal Grammar escapes the net of monadic second order tree logics, and it focuses attention on the mechanisms responsible for a fairly narrow range of empirical data. This is the kind of result that one would hope for in studying the language theoretic complexity of a formalism: that it would serve to carve the theory at the joints and focus attention on particular empirical phenomena and formal mechanisms which fall outside the main body of the framework.

5. Breaking the context free barrier

At the time that the original version of this article appeared in GLOT, the main issue agitating researchers in model theoretic syntax was the problem of the context-free barrier. We have seen that the hierarchy of logics collapses, when applied to trees, at the border of the tree languages strongly generated by context free (string) grammars, in the sense that distinctions between the different tree logics reduce to apparently superficial distinctions in how much memory allocation is hidden in the logic.

The problem which researchers set themselves was not just breaking the context free barrier but remaining decidable in the process. This is a very difficult problem, and it must be admitted right off that it is somewhat artificial in that there is no a priori reason to suppose that natural languages can be described in a decidable logic. The arguments on either side are something like the following. First, the rather slight increases in computational complexity required to get the "mildly context sensitive" languages do suggest that this might be possible. The hunch here would be that the qualities that characterize the mildly context sensitive languages (polynomial parsability, constant growth property) as being like the context-free languages are going to turn out to be reflections of decidability. The problems must not be underestimated, however! It is well known that the monadic second order logic of trees is one of the most powerful decidable logics known. It seems unlikely that any primitive relations can be added to the repertoire of tree description primitives that we have already seen, without making the logic undecidable. Many attempts have been made within logic and all have failed. So it is equally tempting to conjecture that the context-free boundary coincides in some deep sense with the boundary of the decidable logics.

The main results in this area are the following. Kracht (1996) shows that adding a finite number of non-constant formulas to his Constituent Logic would yield descriptions of arbitrary definite-clause grammars, but by the same token that logic is undecidable. Nonetheless, if one is willing to surrender decidability, it offers the possibility of connecting a logical tree oriented principle based specification language to a well understood computational grammar formalism. Given the tight connection between Kracht's propositional dynamic logic and phrase structure grammars, there is no reason why his logic could not by the same token be used as a principle-based specification language for context-free grammars. Seen in this applications-oriented light, the move to formulas with propositional variables may be useful in many applications as a specification language for definite clause grammars. A similar move towards using formal principle based grammars as specifications for more computationally tractable systems is made in Morawietz & Cornell (1997b) where the authors discuss the use of Rogers-style monadic second order logic based grammars as a specification language for tree automata. From the computational perspective, the loss of decidability may be a small price to pay for the possibility of controlling in a systematic way grammars that tend in real applications to be extremely large and unmanageable.

Palm (1997) shows how the set of node labels could be enriched with stacks, yielding an infinite set of node labels and a tree logic sufficient to capture the indexed string languages, but no decidability result is reported for this logic, one way or the other. As with Kracht's results, however, it is important not to overestimate the importance of decidability. There is no guarantee that description languages appropriate to natural language work will be decidable; it could happen that the undecidable questions are questions no one would ever ask, for a trivial example. Nonetheless it remains an interesting question on the border between linguistics and mathematics whether one can cross the context free boundary (or the recognizability boundary, for tree grammars) without losing decidability.

Two research programs that focus on this question are those represented in Mönnich (1997b) and Rogers (1997b). Mönnich (1997b) shows how the complex nonterminals of context free tree grammars could be reified, resulting in recognizable sets of trees which would then yield the underlying context free tree sets upon the application of a rather simple homomorphism. The logic of the reified trees is then decidable, so questions about the original system should be decidable, but the application of a homomorphism makes this a system of two related logics, rather than a single tree logic. Also, because the "lifted" trees look so much different from the trees generated by the original grammar, it is often extremely difficult to reformulate abstract grammatical principles holding of the generated trees so that they now hold of the lifted trees (Kolb 1997). However, Kolb (1997) is able to define the dominance relation holding in the target trees in terms of the structure of the "lifted" trees. Since being able to define dominance entails being able to define c-command, hence presumably intervention and locality, it may be possible to construct principle based grammars defining sets of lifted trees, from which the intended trees can be derived. This would be very useful in a linguistic setting, since it means that questions about the properties of rather more expressive grammars become decidable.

Rogers (Rogers 1998, 1997b) shows how to generalize tree structures to "n-dimensional tree manifolds". Strings are 1-dimenstional tree manifolds, trees are 2-dimenstional tree manifolds. 3-dimensional tree manifolds can be visualized as pyramids in which a node on one floor of the pyramid dominates an entire tree on the floor below it. The results relating structures to automata, closure results on languages and automata and decidability results all extend all the way up this hierarchy of structures, so they have decidable logics at every dimension. There is also a relatively simple connection between 3-dimensional tree manifolds and the derivation structures of tree grammars. Preliminary results relate them to the derivation structures of Tree Adjoining Grammars, and it appears that they can be generalized to the derivation structures of other non-context-free language classes.

References

Aho, A. V. (1968). Indexed grammars—an extension of context-free grammars. *Journal of the ACM* 15, 647–671.

Backofen, R., J. Rogers & K. Vijay-Shanker (1995). A first-order axiomatization of the theory of finite trees. *Journal of Logic, Language and Information* 4, 5–39.

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

Barton, G. E., Jr. & R. C. Berwick (1985). Parsing with assertion sets and information monotonicity. *Proceedings of IJCAI-85.*

Berwick, R. C., S. P. Abney & C. Tenny (1991). *Principle-Based Parsing: Computation and Psycholinguistics.* Dordrecht: Kluwer.

Berwick, R. C. & A. Weinberg (1985). Deterministic parsing: A modern view. *Proceedings of NELS 15.* Brown University, Providence, RI.

Blackburn, P., C. Gardent & W. Meyer-Viol (1993). Talking about trees. *Sixth Conference of the European Chapter of the Association for Computational Linguistics.* EACL.

Blackburn, P. & W. Meyer-Viol (1994). Linguistics, logic, and finite trees. *Bulletin of the IGPL* 2, 2–29.

Blackburn, P., W. Meyer-Viol & M. de Rijke (1995). A proof system for finite trees. Technical Report 67, Computational Linguistics at the University of the Saarland (CLAUS).

Browning, M. A. (1989). ECP $\neq$ CED. *Linguistic Inquiry* 20, 481–491.

Carpenter, B. (1992). *The Logic of Typed Feature Structures: With Applications to Unification Grammars, Logic Programs and Constraint Resolution.* Cambridge: Cambridge University Press.

Chametzky, R. A. (1996). *A Theory of Phrase Markers and the Extended Base.* Albany, NY: State University of New York Press.

Chomsky, N. (1975/1955). *The Logical Structure of Linguistic Theory.* Chicago: University of Chicago Press.

Chomsky, N. (1981). *Lectures on Government and Binding.* Dordrecht: Foris.

Chomsky, N. (1986). *Barriers.* Cambridge, MA: MIT Press.

Chomsky, N. (1995). *The Minimalist Program.* Cambridge, MA: MIT Press.

Cornell, T. L. (1992). *Description Theory, Licensing Theory and Principle-Based Grammars and Parsers.* Ph.D. thesis, UCLA.

Cornell, T. L. (1994). On determining the consistency of partial descriptions of trees. *32nd Annual Meeting of the ACL: Proceedings of the Conference.* Association for Computational Linguistics, Las Cruces, New Mexico.

Cornell, T. L. (1996). A minimalist grammar for the copy language. Arbeitspapiere des Sonderforschungsbereichs 340 79, University of Tübingen.

Cornell, T. L. (1997a). Derivational and representational views of minimalist syntactic calculi. *Proceedings of LACL '97.* Nancy, France. To appear in the Lecture Notes in Artificial Intelligence series by Springer.

Cornell, T. L. (1997b). Representational minimalism. Arbeitspapiere des Sonderforschungsbereichs 340 83, University of Tübingen. Draft of a paper to appear in Kolb and Mönnich, forthcoming.

Doner, J. (1970). Tree acceptors and some of their applications. *Journal of Computer and System Sciences* 4, 406–451.

Engelfriet, J. & E. M. Schmidt (1977). IO and OI. I. *Journal of Computer and System Sciences* 15, 328–353.

Engelfriet, J. & E. M. Schmidt (1978). IO and OI. II. *Journal of Computer and System Sciences* 16, 67–99.

Fischer, M. J. (1968). Grammars with macro-like productions. *Ninth Annual Symposium on Switching and Automata Theory*. IEEE.

Fong, S. (1992). *Computational Properties of Principle-Based Grammatical Theories*. Ph.D. thesis, MIT.

Frank, R. (1990). *Computation and Linguistic Theory: A Government Binding Theory Parser Using Tree Adjoining Grammar*. Master's thesis, University of Pennsylvania.

Frank, R. (1992). *Syntactic Locality and Tree Adjoining Grammar: Grammatical, Acquisition and Processing Perspectives*. Ph.D. thesis, University of Pennsylvania.

Frank, R. & K. Vijay-Shanker (1995). C-command and grammatical primitives. *18th GLOW Colloquium—Univ. of Tromsø*.

Frank, R. & K. Vijay-Shanker (1998). TAG derivation as monotonic c-command. *Proceedings of the Fourth Workshop on Tree-Adjoining Grammars and Related Frameworks TAG+98*. Philadelphia, PA.

Gärtner, H.-M. (1997). *Generalized Transformations and Beyond*. Ph.D. thesis, Johann-Wolfgang-Goethe Universität zu Frankfurt am Main.

Gécseg, F. & M. Steinby (1984). *Tree Automata*. Budapest: Akadémiai Kiadó.

Gécseg, F. & M. Steinby (1997). Tree languages. *Handbook of Formal Languages: Beyond Words*, edited by G. Rozenberg & A. Salomaa, volume 3. Berlin: Springer.

Gibson, E. (1991). *A Computational Theory of Human Linguistic Processing*. Ph.D. thesis, Carnegie-Mellon.

Gibson, E. (1995). *Memory Limitations and Sentence Processing Breakdown*. Cambridge, MA: MIT Press.

Gorrell, P. (1991). Subcategorization and sentence processing. *Principle-Based Parsing: Computation and Psycholinguistics*, edited by R. Berwick, S. Abney & C. Tenney. Dordrecht: Kluwer.

Gorrell, P. (1995). *Syntax and Parsing.* Cambridge: Cambridge University Press.

Hopcroft, J. E. & J. D. Ullman (1979). *Introduction to Automata Theory, Languages and Computation.* Addison-Wesley.

Huybregts, R. (1984). The weak inadequacy of context-free phrase structure grammars. *Van Periferie naar Kern*, edited by de Haan, Trommelen & Zonneveld. Dordrecht: Foris, 81–99.

Johnson, M. (1991). Deductive parsing: The use of knowledge of language. *Principle-Based Parsing: Computation and Psycholinguistics*, edited by R. C. Berwick, S. P. Abney & C. Tenny. Dordrecht: Kluwer, 39–64.

Johnson, M. (1994). Two ways of formalizing grammars. *Linguistics and Philosophy* 17, 221–248.

Joshi, A. K., K. Vijay-Shanker & D. J. Weir (1991). The convergence of mildly context-sensitive grammatical formalisms. *Foundational Issues in Natural Language Processing*, edited by P. Sells, S. Shieber & T. Wasow. Cambridge, MA: MIT Press, 31–81.

Kayne, R. S. (1994). *The Antisymmetry of Syntax.* Cambridge, MA: MIT Press.

Keenan, E. L. & E. P. Stabler (1992). Language invariants. *Proceedings of the Eighth Amsterdam Colloquium*, edited by P. Dekker & M. Stokhof.

Keenan, E. L. & E. P. Stabler (1995). There is more than one language. *Proceedings: Langues et Grammaires-1.* Paris.

Keller, B. (1993). *Feature Logics, Infinitary Descriptions and Grammar*, volume 44 of *Lecture Notes.* Stanford: CSLI.

Kolb, H.-P. (1997). Macros for minimalism: Towards weak descriptions of strong structures. Arbeitspapiere des Sonderforschungsbereichs 340 113, University of Tübingen. Draft of a paper to appear in Kolb and Mönnich, forthcoming.

Kolb, H.-P. & U. Mönnich (forthcoming). *Mathematics of Syntactic Structure: Trees and Their Logics.* Mouton-de Gruyter.

Kracht, M. (1992). The theory of syntactic domains. Technical Report 75, Department of Philosophy, University of Utrecht.

Kracht, M. (1993a). Mathematical aspects of command relations. *Sixth Conference of the European Chapter of the Association for Computational Linguistics.* EACL.

Kracht, M. (1993b). Nearness and syntactic influence spheres. Ms., II. Mathematisches Institut, Freie Universität Berlin.

Kracht, M. (1995a). Is there a genuine modal perspective on feature structures? *Linguistics and Philosophy* 18, 401–458.

Kracht, M. (1995b). Syntactic codes and grammar refinement. *Journal of Logic, Language and Information* 4, 41–60.

Kracht, M. (1996). On reducing principles to rules. *Specifying Syntactic Structures*, edited by P. Blackburn & M. de Rijke. Stanford, CA: CSLI Publications.

Kracht, M. (1997). Inessential features. *Proceedings of the 1st International Conference on Logical Aspects of Computational Linguistics (LACL-96)*, edited by C. Retoré, volume 1328 of *LNAI*. Berlin: Springer.

Kroch, T. & O. Rambow (1994). The linguistic relevance of quasi-trees. *3e Colloque International sur les Grammaires d'Arbres Adjoints (TAG+3)*. Technical Report TALANA-RT-94-01, TALANA, Université Paris 7.

Lasnik, H. & J. J. Kupin (1977). A restrictive theory of transformational grammar. *Theoretical Linguistics* 4, 173–196.

Lecomte, A. (1997). POM-nets and minimalism. *Proceedings of the IV Roma Workshop: Dynamic Perspectives in Logic and Linguistics*, edited by C. Casadio. SILPS Group in Logic and Natural Languages.

Macías, B. (1990). *An Incremental Parser for Government-Binding Theory.* Ph.D. thesis, University of Cambridge.

Maibaum, T. S. E. (1974). A generalized approach to formal languages. *Journal of Computer and System Sciences* 8, 409–439.

Manzini, M. R. (1992). *Locality: A Theory and Some of Its Empirical Consequences.* Cambridge, MA: MIT Press.

Marcus, M. & D. Hindle (1990). Description theory and intonation boundaries. *Cognitive Models of Speech Processing*, edited by G. T. M. Altman. MIT Press, 483–512.

Marcus, M. P. (1987). Deterministic parsing and description theory. *Linguistic Theory and Computer Applications*, edited by P. Whitelock, M. M. Wood, H. L. Somers, R. Johnson & P. Bennet. New York: Academic Press.

Marcus, M. P., D. Hindle & M. M. Fleck (1983). D-theory: Talking about talking about trees. *21st Annual Meeting of the ACL: Proceedings of the Conference.*

McCawley, J. D. (1968). Concerning the base component of a transformational grammar. *Foundations of Language* 4, 243–269.

Mönnich, U. (1997a). Adjunction as substitution: An algebraic formulation of regular, context-free and tree adjoining languages. *Formal Grammar 1997: Proceedings of the Conference*, edited by G.-J. M. Kruijff, G. V. Morrill & R. T. Oehrle.

Mönnich, U. (1997b). On cloning context-freeness. Arbeitspapiere des Sonderforschungsbereichs 340 114, University of Tübingen. Draft of a paper to appear in Kolb and Mönnich, forthcoming.

Moortgat, M. (1996). Categorial type logics. *Handbook of Logic and Language*, edited by J. van Benthem & A. ter Meulen, chapter 2. Elsevier.

Morawietz, F. & T. Cornell (1997a). On the recognizability of relations over a tree definable in a monadic second order tree description language. Arbeitspapiere des Sonderforschungsbereichs 340 85, University of Tübingen.

Morawietz, F. & T. Cornell (1997b). Representing constraints with automata. *35th Annual Meeting of the ACL: Proceedings of the Conference*. Association for Computational Linguistics, Madrid, Spain.

Morrill, G. (1994). *Type Logical Grammar*. Dordrecht: Kluwer.

Palm, A. (1997). *Transforming Tree Constraints into Formal Grammars: The Expressivity of Tree Languages*, volume 173 of *Dissertationen zur künstlichen Intelligenz*. Sankt Augustin, Germany: Infix.

Partee, B. H., A. ter Meulen & R. E. Wall (1990). *Mathematical Methods in Linguistics*. Dordrecht: Kluwer.

Peters, P. S. & R. W. Ritchie (1973). Context-sensitive immediate constituent analysis: Context-free languages revisited. *Mathematical Systems Theory* 6, 324–333.

Rabin, M. O. (1969). Decidability of second-order theories and automata on infinite trees. *Transactions of the American Mathematical Society* 141, 1–35.

Rambow, O. (1994a). *Formal and Computational Models for natural Language Syntax*. Ph.D. thesis, University of Pennsylvania.

Rambow, O. (1994b). Multiset-valued linear index grammars: Imposing dominance constraints on derivations. *32nd Annual Meeting of the ACL: Proceedings of the Conference*. Association for Computational Linguistics.

Richardson, J. & R. Chametzky (1985). A string based reformulation of c-command. *Proceedings of NELS 15*.

Rizzi, L. (1990). *Relativized Minimality*. Cambridge, Mass.: MIT Press.

Rodman, R. (1977). Axiomatizing linguistic structures. *Theoretical Linguistics* 4, 255–264.

Rogers, J. (1994). *Studies in the Logic of Trees with Applications to Grammar Formalisms*. Ph.D. thesis, University of Delaware, Department of Computer & Information Sciences.

Rogers, J. (1997a). On descriptive complexity, language complexity, and GB. *Specifying Syntactic Structures*, edited by P. Blackburn & M. de Rijke. Cambridge: CSLI/Cambridge University Press.

Rogers, J. (1997b). A unified notion of derived and derivation structures in TAG. *Proceedings of the Fifth Meeting on Mathematics of Language (MOL5)*, edited by T. Becker & H.-U. Krieger. Saarbrücken: DFKI.

Rogers, J. (1998). On defining TALs with logical constraints. *Proceedings of the Fourth Workshop on Tree-Adjoining Grammars and Related Frameworks TAG+98*. Philadelphia.

Rogers, J. (forthcoming). *A Descriptive Approach to Language-Theoretic Complexity.* Studies in Logic, Language and Information. Cambridge: CSLI/Cambridge Univ. Press.

Rogers, J. & K. Vijay-Shanker (1992). Reasoning with descriptions of trees. *30th Annual Meeting of the ACL: Proceedings of the Conference*. Association for Computational Linguistics.

Rogers, J. & K. Vijay-Shanker (1995). Obtaining trees from their descriptions: An application to tree-adjoining grammars. *Computational Intelligence* 10, 401–421.

Rounds, W. C. (1970). Mappings and grammars on trees. *Mathematical Systems Theory* 4, 257–287.

Shieber, S. M. (1985). Evidence against the context-freeness of natural language. *Linguistics and Philosophy* 8, 333–343.

Stabler, E. (1994). The finite connectivity of linguistic structure. *Perspectives on Sentence Processing*, edited by C. C. Jr., L. Frazier & K. Rayner. Hillsdale, New Jersey: Lawrence Erlbaum, 303–336.

Stabler, E. (1997). Derivational minimalism. *Logical Aspects of Computational Linguistics*, edited by C. Retoré. Berlin: Springer. LNAI 1328.

Stabler, E. P. (1992). *The Logical Approach to Syntax: Foundations, Specifications, and Implementations of Theories of Government and Binding*. Cambridge, MA: MIT Press.

Sturt, P. & M. Crocker (1996). Monotonic syntactic processing: A cross-linguistic study of attachment. *Language and Cognitive Processes* 11, 449–494.

Thatcher, J. W. & J. B. Wright (1968). Generalized finite automata theory with an application to a decision problem of second-order logic. *Mathematical Systems Theory* 2, 57–81.

Thomas, W. (1997). Languages, automata, and logic. *Handbook of Formal Languages: Beyond Words*, edited by G. Rozenberg & A. Salomaa, volume 3. Berlin: Springer.

Vijay-Shanker, K. (1992). Using descriptions of trees in a tree adjoining grammar. *Computational Linguistics* 18, 481–517.

Volger, H. (1997). Principle languages and principle based parsing. Arbeitspapiere des Sonderforschungsbereichs 340 82, University of Tübingen. Draft of a paper to appear in Kolb and Mönnich, forthcoming.

Wall, R. (1972). *Introduction to Mathematical Linguistics.* Prentice-Hall.

Weinberg, A. (1988). *Locality Principles in Syntax and in Parsing.* Ph.D. thesis, MIT.

Weinberg, A. (1994). Parameters in the theory of sentence processing: Minimal commitment theory goes east. *Journal of Psycholinguistic Research* 22, 339–364.