Some Interdefinability Results for Syntactic Constraint Classes

Thomas Graf

Department of Linguistics
University of California, Los Angeles
tgraf@ucla.edu
http://tgraf.bol.ucla.edu

Abstract. Choosing as my vantage point the linguistically motivated Müller-Sternefeld hierarchy [23], which classifies constraints according to their locality properties, I investigate the interplay of various syntactic constraint classes on a formal level. For non-comparative constraints, I use Rogers's framework of multi-dimensional trees [31] to state Müller and Sternefeld's definitions in general yet rigorous terms that are compatible with a wide range of syntactic theories, and I formulate conditions under which distinct non-comparative constraints are equivalent. Comparative constraints, on the other hand, are shown to be best understood in terms of optimality systems [5]. From this I derive that some of them are reducible to non-comparative constraints. The results jointly vindicate a broadly construed version of the Müller-Sternefeld hierarchy, yet they also support a refined picture of constraint interaction that has profound repercussions for both the study of locality phenomena in natural language and how the complexity of linguistic proposals is to be assessed.

Key words: Syntactic constraints, Transderivationality, Economy conditions, Model theoretic syntax, Multi-dimensional trees, Optimality systems

Introduction

Constraints are arguably one of the most prominent tools in modern syntactic analysis. Although the dominance of derivational approaches in the linguistic mainstream since the inception of Chomsky's Minimalist Program [2, 3] might suggest otherwise, generative frameworks still feature a dazzling diversity of principles and well-formedness conditions. The array of commonly assumed constraints ranges from the well-established Shortest Move Constraint to the fiercely debated principles of binding theory, but we also find slightly more esoteric proposals such as Rule I [26], MaxElide [35], GPSG's Exhaustive Constant Partial Ordering Axiom [6] or the almost forgotten Avoid Pronoun Principle of classic GB. A closer examination of these constraints shows that they differ significantly in the structures they operate on and how they succeed at restricting the set of expressions. A natural question to ask, then, is if we can identify commonalities between different constraints, and what the formal and linguistic content of these commonalities might be.

The Müller-Sternefeld (MS) hierarchy [21, 23] is — to my knowledge — the only articulate attempt at a classification of linguistic constraints so far. Basing their analysis on linguistic reasoning grounded in locality considerations, Müller and Sternefeld distinguish several kinds of constraints, which in turn can be grouped into two bigger classes. The first one is the class of *non-comparative* constraints (NCCs): *representational* constraints are well-formedness conditions on standard trees (e.g. ECP, government), *derivational* constraints restrict the shape of trees that are adjacent in a derivation (e.g. Shortest Move), and *global* constraints apply to derivationally non-adjacent trees (e.g. Projection Principle). The second class is instantiated by *comparative* constraints (CCs), which operate on sets of structures. Given a set of structures, a CC returns the best member(s) of this set, which is usually called the optimal candidate. Crucially, the optimal candidate does not have to be well-formed — it just has to be better than the competing candidates. Müller slightly revises this picture in [21] and further distinguishes CC according to the type of structures they operate on. If the structures in question are trees, the constraint is called *translocal* (e.g. Avoid Pronoun Principle); if they are derivations, it is called *transderivational* (e.g. Fewest Steps, MaxElide, Rule I). Finally, it is also maintained in [21] that these five subclasses can be partially ordered by their expressivity: representational = derivational < global < translocal < transderivational. A parametric depiction of the constraint classification and the expressivity hierarchy, which jointly make up the MS-hierarchy, is given in Fig. 1.

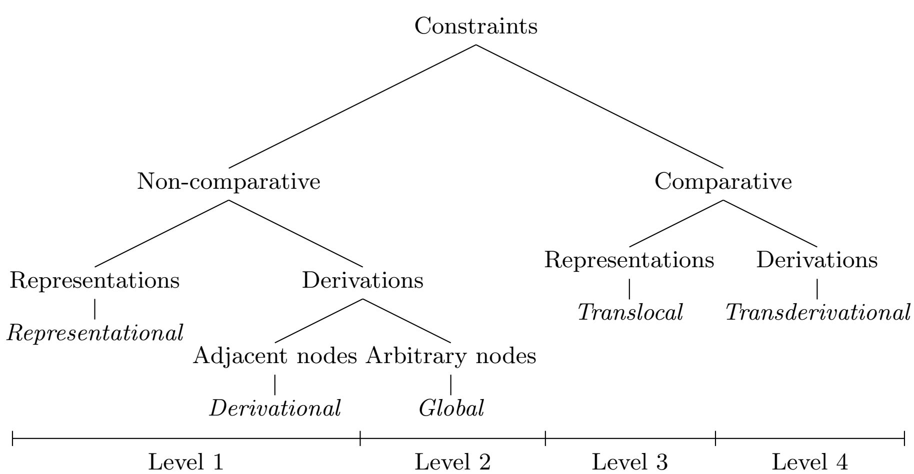


Fig. 1. The Müller-Sternefeld hierarchy of constraints

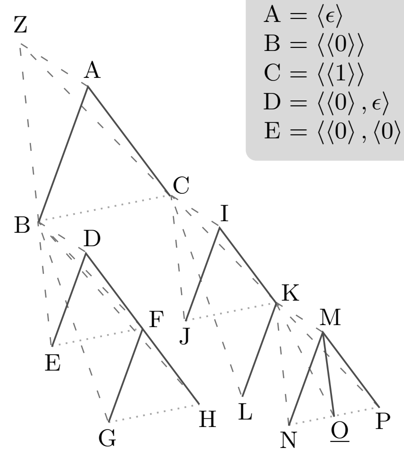
The MS-hierarchy has a strong intuitive appeal, at least insofar as derivations, long-distance restrictions and operations on sets seem more complex than representations, strictly local restrictions and operations on trees, respectively. However, counterexamples are readily at hand. For instance, it is a simple coding exercise to implement any transderivational constraint as a global constraint by concatenating the distinct derivations into one big derivation, provided there are no substantial restrictions on how we may enrich our grammar formalism. As another example, it was shown in [16] that Minimalist Grammars with the Specifier Island Constraint (SPIC) but without the Shortest Move Constraint can generate any type-0 language. But the SPIC is a very simple derivational constraint, so it unequivocally belongs to the weakest class in the hierarchy, which is at odds with its unexpected effects on expressivity. Therefore, the MS-hierarchy makes the wrong predictions in its current form, or rather, it makes no predictions at all, because its notion of complexity and its assumptions concerning the power of the syntactic framework are left unspecified.

In this paper, I show how a model theoretically informed perspective does away with these shortcomings and enables us to refine the MS-hierarchy such that the relations between constraint classes can be studied in a rigorous yet linguistically insightful way. In particular, I adopt Rogers's multi-dimensional trees framework [31] as a restricted metatheory of linguistic proposals in order to ensure that the results hold for a wide range of syntactic theories. We proceed as follows: After a brief discussion of technical preliminaries I move on to the definition of classes of NCCs in Sect. 2 and study their behavior and interrelationship in arbitrary multi-dimensional tree grammars. I show that a proper subclass of the global constraints can be reduced to local constraints. In Sect. 3, I then turn to a discussion of CCs, why they require the model theoretic approach to be supplemented by optimality systems [5], and which CCs can be reduced to NCCs.

1 Preliminaries

Most of my results I couch in terms of the multi-dimensional tree (MDT) framework developed by Rogers [31, 32]. The main appeal of MDTs for this endeavor is that they make it possible to abstract away from theory-specific idiosyncrasies. This allows for general characterizations of constraint classes and their reducibility that hold for a diverse range of linguistic theories. MDT renditions of GB [29], GPSG [27] and TAG [30] have already been developed; the translation procedure from HPSG to TAG defined in [14] should allow us to reign in (a fragment of) the former as well. Further, recent results suggest that an approximation of Minimalist Grammars [33] is feasible, too: for every Minimalist Grammar we can construct a strongly equivalent k-MCFG [20], and for each $k \geq 2$, the class of 2^{k-1}-MCFLs properly includes the class of level-k control languages [12], which in turn are equivalent to the string yield of the set of $(k+1)$-dimensional trees [31].

While initially intimidating due to cumbersome notation, MDTs are fairly easy to grasp at an intuitive level. Looking at familiar cases first, we note that a string can be understood as a unary branching tree, a set of nodes ordered by



Z = ϵ	F = $\langle\langle 0\rangle, \langle 1\rangle\rangle$	L = $\langle\langle 1\rangle, \langle 1, 0\rangle\rangle$
A = $\langle\epsilon\rangle$	G = $\langle\langle 0\rangle, \langle 1, 0\rangle\rangle$	M = $\langle\langle 1\rangle, \langle 1\rangle, \epsilon\rangle$
B = $\langle\langle 0\rangle\rangle$	H = $\langle\langle 0\rangle, \langle 1, 1\rangle\rangle$	N = $\langle\langle 1\rangle, \langle 1\rangle, \langle 0\rangle\rangle$
C = $\langle\langle 1\rangle\rangle$	I = $\langle\langle 1\rangle, \epsilon\rangle$	O = $\langle\langle 1\rangle, \langle 1\rangle, \langle 1\rangle\rangle$
D = $\langle\langle 0\rangle, \epsilon\rangle$	J = $\langle\langle 1\rangle, \langle 0\rangle\rangle$	P = $\langle\langle 1\rangle, \langle 1\rangle, \langle 2\rangle\rangle$
E = $\langle\langle 0\rangle, \langle 0\rangle\rangle$	K = $\langle\langle 1\rangle, \langle 1\rangle\rangle$	

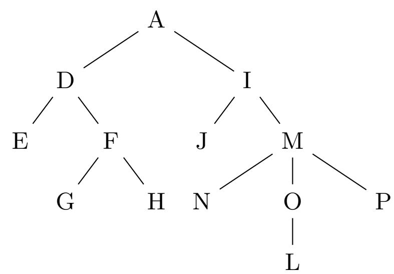


Fig. 2. A T^3 (with O a foot node), its node addresses and its 2-dimensional yield

the precedence relation. But as there is only one axis along which its nodes are ordered, it is reasonable to call a string a one-dimensional tree, rather than a unary branching one. In a standard tree, on the other hand, the set of nodes is ordered by two relations, usually called dominance and precedence. Suppose s is the mother of two nodes t and u in some standard tree, and also assume that t precedes u. Then we might say that s dominates the string tu. Given our new perspective on strings as one-dimensional trees, this suggests to construe standard trees as relating nodes to one-dimensional trees by immediate dominance. Thus it makes only sense to refer to them as two-dimensional objects. But from here it is only a small step to the concept of MDTs. A three-dimensional tree (see Fig. 2 for an example) relates nodes to two-dimensional, i.e. standard trees (for readers familiar with TAG, it might be helpful to know that three-dimensional trees correspond to TAG derivations). A four-dimensional tree relates nodes to three-dimensional trees, and so on. In general, a d-dimensional tree is a set of nodes ordered by d dominance relations such that the n^{th} dominance relation relates nodes to $(n-1)$-dimensional trees (for $d = 1$, assume that single nodes are zero-dimensional trees).

To make this precise, we define d-dimensional trees as generalizations of Gorn tree domains. First, let a *higher-order sequence* be defined inductively as follows:

- ${}^{0}1 := \{1\}$
- ${}^{n+1}1$ is the smallest set containing $\langle\rangle$ and if both $\langle x_1, \ldots, x_l\rangle \in {}^{n+1}1$ and $y \in {}^{n}1$ then $\langle x_1, \ldots, x_l, y\rangle \in {}^{n+1}1$.

Concatenation of sequences is denoted by $\cdot$ and defined only for sequences of the same order. A 0-dimensional tree is either $\emptyset$ or $\{1\}$. For $d \geq 1$, a *d-dimensional tree* T^d is a set of d^{th}-order sequences satisfying

- $T^d \subseteq {}^d1$, and
- $\forall s, t \in {}^d1[s \cdot t \in T^d \rightarrow s \in T^d]$, and
- $\forall s \in {}^d1[\{w \in {}^{(d-1)}1 \mid s \cdot \langle w \rangle \in T^d\}$ is a $(d-1)$-dimensional tree].

The reader might want to take a look at Fig. 2 again for a better understanding of the correspondence between sequences and tree nodes (first-order sequences are represented by numerals to improve readability; e.g. $0 = \langle\rangle$ and $2 = \langle 1, 1\rangle$).

Several important notions are straightforwardly defined in terms of higher-order sequences. The *leaves* of T^d are the nodes at addresses that are not properly extended by any other address in T^d. The *depth* of T^d is the length of its longest top level sequence, which in more intuitive terms corresponds to the length of the longest path of successors at dimension d from the root to a leaf. Given a T^d and some node s of T^d, the *child structure* of s in T^d is the set $\{t \in T^{d-1} \mid s \cdot \langle t \rangle \in T^d\}$. For example, the child structure of B in Fig. 2 is the T^2 with its root labeled D. For any T^d and $1 \leq i \leq d$, its *branching factor at dimension* i is 1 plus the maximum depth of the T^{i-1} child structures contained by T^d. If the branching factor of some T^d is at most n for all dimensions $1 \leq i \leq d$, we call it n*-branching* and write T^d_n.

For any non-empty alphabet Σ, $\mathcal{T}^d_\Sigma := \langle T, \ell \rangle$, T a T^d and ℓ a function from Σ to $\wp(T)$, is a Σ*-labeled* d-dimensional tree. In general, we require all trees to be labeled and simply write T^d_Σ. The i*-dimensional yield* of T^d_Σ is obtained by recursively rewriting all nodes at dimension $j > i$, starting at dimension d, by their $(j-1)$-dimensional child structure. Trees with more than two dimensions have some of their leaves at each dimension $i > 2$ marked as foot nodes, which are the joints where the $(i-1)$ child structures are merged together. In forming the 2-dimensional yield of our example tree, K is rewritten by the 2-dimensional tree rooted by M. The daughter of K ends up dominated by O rather than N or P because O is marked as the foot node. For a sufficiently rigorous description of how the i-dimensional yield is computed, see [31, p.281–283] and [32, p.301–307]. A sequence $\langle s_1, \ldots, s_m \rangle$ of nodes of T^d, $m \geq 1$, is an i*-path* iff with respect to the i-dimensional yield of T^d, s_1 is the root, s_m a leaf, and for all s_j, s_{j+1}, $1 \leq j < m$, it holds that s_j immediately dominates s_{j+1} at dimension i. The set of all i-paths of T^d is its i*-path language*.

A set of T^d_Σs is also called a T^d *language*, denoted L^d_Σ. Unless stated otherwise, the branching factor is assumed to be bounded for every L^d_Σ, that is to say, there is some $n \in \mathbb{N}$ such that each $T \in L^d_\Sigma$ is n-branching. Call T^d *local* iff its depth is 1. In Fig. 2, the T^3 rooted by K and the T^2 rooted by M are local; the T^3 rooted by B is also local, even though its child structure, the T^2 rooted by D, is not. A T^d *grammar* $\mathcal{G}^d_\Sigma$ over an alphabet Σ is a finite language of local T^d_Σs. Let $\mathcal{G}^d_\Sigma(\Sigma_0)$ denote the set of T^d_Σs licensed by a grammar $\mathcal{G}^d_\Sigma$ relative to a set of initial symbols $\Sigma_0 \subseteq \Sigma$, which is the set of all T^d_Σs with their root labeled by a symbol drawn from Σ_0 and each of their local d-dimensional subtrees contained in $\mathcal{G}^d_\Sigma$. A language L^d_Σ is a *local set* iff it is $\mathcal{G}^d_\Sigma(\Sigma_0)$ for some $\mathcal{G}^d_\Sigma$ and some $\Sigma_0 \subseteq \Sigma$. Intuitively, a local set of T^ds is a T^d language where all trees can be built up from local trees. An important fact about local sets is that they are fully characterized by *subtree substitution closure*.

Theorem 1 (Subtree substitution closure). *L^d_Σ is a local set of T^d_Σs iff for all $T, T' \in L^d_\Sigma$, all $s \in T$ and all $t \in T'$, if s and t have the same label, then the result of substituting the subtree rooted by s for the subtree rooted by t is in L^d_Σ.*

Proof. An easy lift of the proof in [28] to arbitrary dimension d. □

For our logical approach, we interpret a $T^d_{n,\Sigma}$ as an initial segment of the relational structure $\mathfrak{T}^d_n := \left\langle \mathsf{T}^d_n, \lhd_i \right\rangle_{1 \leq i \leq d}$, where T^d_n is the infinite T^d in which every point has a child structure of depth $n-1$ in all its dimensions, and where $\lhd_i$ denotes immediate dominance at dimension i, that is $x \lhd_i y$ iff y is the immediate successor of x in the i^{th} dimension.

$$
\begin{aligned}
x \lhd_d y &\text{ iff } y = x \cdot \langle s \rangle \\
x \lhd_{d-1} y &\text{ iff } x = p \cdot \langle s \rangle \text{ and } y = p \cdot \langle s \cdot \langle w \rangle \rangle \\
&\vdots \\
x \lhd_1 y &\text{ iff } x = p \cdot \langle s \cdot \langle \cdots \langle w \rangle \cdots \rangle \rangle \text{ and } y = p \cdot \langle s \cdot \langle \cdots \langle w \cdot 1 \rangle \cdots \rangle \rangle
\end{aligned}
$$

The weak monadic second-order logic for $\mathfrak{T}^d_n$ is denoted by MSO^d and includes — besides the usual connectives, quantifiers and grouping symbols — constants for each $\lhd_i$, $1 \leq i \leq d$, and two countably infinite sets of variables ranging over individuals and finite subsets, respectively. As usual, we write $\mathfrak{T}^d_n \models \phi[\mathbf{s}]$ to assert that ϕ is satisfied in $\mathfrak{T}^d_n$ under assignment $\mathbf{s}$. For any T^d, all quantifiers are assumed to be implicitly restricted to the initial segment of $\mathfrak{T}^d_n$ corresponding to T^d. The set of models of ϕ is denoted by $\textsc{Mod}(\phi)$. This notation extends to sets of formulas in the obvious way. Note that L^d_Σ is *recognizable* iff $L^d_\Sigma = \textsc{Mod}(\Phi)$ for some set Φ of MSO^d formulas.

Let me close this section with several minor remarks. The notation $A \setminus B$ is used to denote set difference. Regular expressions are employed at certain points in the usual way, with the small addition of $x^{\leq 1}$ as a stand-in for ϵ and x. Finally, I will liberally drop subscripts and superscripts whenever possible.

2 Non-comparative Constraints

2.1 Logics for Non-comparative Constraints

A short glimpse at the MS-hierarchy in Fig. 1 reveals that NCCs are distinguished by two parameters: the distance between the nodes they restrict (1 versus unbounded) and the type of structure they operate on (representations versus derivations). As I will show now, this categorization can be sharpened by recasting it in logical terms, thereby opening it up to our mathematical explorations in the following section.

The distinction between representations and derivations is merely a terminological confusion in our multi-dimensional setup. A two-dimensional tree, for instance, can be interpreted as both a representational tree structure and a string derivation. This ambiguity is particularly salient for higher dimensions, where there are no linguistic preconceptions concerning the type of structure we are operating on. A better solution, then, is to distinguish NCCs according to the highest dimension they mention in their specification (this will be made precise soon).

As for the distance between restricted nodes, it seems to be best captured by the distinction between local and recognizable sets, the latter allowing for unbounded dependencies between nodes while the former are limited to well-formedness conditions that apply within trees of depth 1. As mentioned in Sect. 1, definability in MSO is a logical characterization of recognizability, so in conjunction with the MDT framework, this already gives us everything we need to give a theory-neutral definition of global NCCs. For the second, restricted kind of constraints, however, we still need a logical characterization of local sets. Fortunately, this characterization was obtained for two-dimensional trees by Rogers in [28] and can easily be lifted to higher dimensions as follows.

For any $D \in \{\triangleleft_i, \triangleright_i\}_{i \geq 1}$, let $D\phi(x)$ abbreviate the MSO^k formula $\exists y[xDy \wedge \phi(y)]$, where $x \triangleright_i y := y \triangleleft_i x$. We require that $\mathfrak{T}^d_n \models D\phi(x)[\mathbf{s}]$ iff $\mathfrak{T}^d_n \models \forall x \exists y[xDy \wedge \phi(y)][\mathbf{s}]$. Declaring all other uses of quantification to be illicit yields what may be regarded as a normal modal logic.

Definition 2 (RLOC^k). *RLOC^k (relaxed LOC^k) is the smallest set of MSO^k formulas over the boolean operators, individual variables, set variables and all $\triangleleft_i$, $\triangleright_i$, $1 \leq i \leq k$.*

In the next step, we restrict disjunction. Let LOC^k_+ be the smallest set of RLOC^k formulas such that

- all $\triangleleft_i$ and $\triangleright_j$, $i < j \leq k$, are in the scope of exactly one more $\triangleleft_k$ than $\triangleright_k$, and
- all $\triangleleft_k$ are in the scope of exactly as many $\triangleright_k$ as $\triangleleft_k$.

Similarly, let LOC^k_- be the smallest set of RLOC^k formulas such that

- all $\triangleleft_i$ and $\triangleright_j$, $i < j \leq k$, are in the scope of exactly as many $\triangleleft_k$ as $\triangleright_k$, and
- all $\triangleleft_k$ are in the scope of exactly one more $\triangleright_k$ than $\triangleleft_k$.

Definition 3 (LOC^k). *The set of LOC^k formulas consists of all and only those formulas that are conjunctions of*

- *disjunctions of formulas in LOC^k_+, and*
- *disjunctions of formulas in LOC^k_-.*

The following lemmata tell us that LOC^k restricts only $\triangleleft_k$ and $\triangleright_k$ in a meaningful way. This will also be of use in the next section.

Lemma 4 (RLOC^k and LOC^{k+1}). *A formula ϕ is an RLOC^k formula iff it is a LOC^{k+1} formula containing no $\triangleleft_{k+1}$ and no $\triangleright_{k+1}$.*

Proof. By induction on the complexity of ϕ. The crucial condition is the first clause in the definition of LOC^k_-. □

Lemma 5 (Normal forms). *Every LOC^k_+ formula is equivalent to a disjunction of conjunctions of LOC^k_+ formulas of the form $(\triangleleft_k\{\triangleleft_i, \triangleright_i\}^*_{1 \leq i < k})^{\leq 1}\phi$, ϕ a propositional formula. Similarly, every LOC^k_- formula is equivalent to a disjunction of conjunctions of LOC^k_- formulas of the form $(\{\triangleleft_i, \triangleright_i\}^*_{1 \leq i < k}\triangleright_k)^{\leq 1}\phi$.*

Proof. The proof in [28] holds for all $k \geq 1$. □

With Lemma 5 under our belt, we can proceed to prove the sought after equivalence of definability in LOC^d and locality of sets of d-dimensional trees.

Theorem 6 (Locality and LOC). *A set L of finite T_n^ds, $d, n \geq 1$, is local iff it is definable in LOC^d.*

Proof. As the proof in [28] for the correspondence between LOC^2 and the local sets of 2-dimensional trees is easily generalized to all positive $d \neq 2$, a short sketch suffices.

⇒ Since L is local, there is a grammar $\mathcal{G}$ that derives L, i.e. L can be fully specified by a finite set of trees of depth 1. Assume that $T_1, \ldots, T_n$ are all the trees in $\mathcal{G}$ with their root labeled A and $\phi_1, \ldots, \phi_n$ are RLOC^{d-1} formulas describing the child structure of A in $T_1, \ldots, T_n$, respectively (for $d = 1$, ϕ_i is propositional). As there is an upper bound on the size of child structures for all $T \in \mathcal{G}$, such ϕ are guaranteed to exist. Then $\phi_A := A \rightarrow \lhd_d \bigvee_{1 \leq i \leq n} \phi_i$ is a LOC_+^d formula, whence $\phi_\Sigma := \bigwedge_{A \in \Sigma} \phi_A$ is in LOC^d. It only remains to conjoin ϕ_Σ with the LOC_-^d formulas $\bigvee_{A \in \Sigma} A(x)$ and $\neg \rhd_d \top \rightarrow \bigvee_{A \in \Sigma_0} A(x)$ to ensure that all nodes are labeled and in particular that root nodes are labeled with an initial symbol. The result is a LOC^d formula.

⇐ This follows from Lemma 5. It is easy to see that the truth value of LOC formulas in normal form at some node t depends only on the local tree rooted at either some t' in the same local tree as t for LOC_+^k or the parent of t' for LOC_-^k. In either case the truth value of the formula remains unaffected by subtree substitution, whence all sets satisfying it are local. □

Thus everything is in place now for the logical classification of NCCs we outlined before.

Definition 7 (Classes of non-comparative constraints). *A constraint c is*

- k-global *iff it can be defined by an MSO^k formula.*
- k-local *iff it can be defined by a LOC^k formula.*
- fully k-local *iff*
  - *for $k = 1$, c is 1-local*
  - *for $k > 1$, c is definable by a LOC^k formula ϕ built up from LOC_+^k and LOC_-^k formulas $\phi_1, \ldots, \phi_n$ in normal form such that for each $1 \leq i \leq n$ the formula ψ_i obtained from ϕ_i by removing all occurrences of $\lhd_k$ and $\rhd_k$ is fully $(k-1)$-local.*

2.2 Reducibility with and without Fixed Signatures

We now turn to interdefinability results for NCCs. The well-understood relation between local and recognizable sets [4, 36] in conjunction with their logical definability [28, 31] immediately derives the reducibility of global constraints.

Theorem 8 (Reducibility by features). *Let Φ be a set of MSO^d formulas and c_g a k-global constraint, $1 \leq k \leq d$, with $\text{Mod}(\Phi \cup \{c_g\})$ a recognizable set of Σ-labeled T^ds. Then there is a fully k-local constraint c_l such that $\text{Mod}(\Phi \cup \{c_l\})$ is a set of $\Sigma \cup \Omega$-labeled T^ds and a projection of $\text{Mod}(\Phi \cup \{c_l\})$.*

The familiar idea underlying the theorem is that we only need to set aside a certain amount of diacritic features to make all the non-local information used in c_g accessible to c_l. The details of this procedure were studied by Marcus Kracht in his work on coding theory [17–19].[1]

Unfortunately, Theorem 8 is at most of peripheral importance to linguists, who usually do not want their grammar to contain spurious labels or features that have no independent empirical motivation. But comparable results can be obtained if the signature is fixed, at least for all dimensions but the highest one. The trick is to exploit the structure of trees to reencode global constraints as local constraints at higher dimensions. Lemma 4 already hinted at this possibility, but it is too weak to actually derive it. The missing piece of the puzzle is the expressivity of LOC^d with respect to RLOC^{d-1} and MSO^{d-1} at dimension $d-1$, which is partially answered by the following two lemmata.

Lemma 9 ($\mathbf{RLOC}^{d-1} < \mathbf{LOC}^d$). *There is a set Φ of LOC^d formulas, $d > 1$, such that the $(d-1)$-dimensional yield of $\text{Mod}(\Phi)$ is not definable in RLOC^{d-1}.*

Proof. We already know from Lemma 4 that $\text{RLOC}^{d-1} \leq \text{LOC}^d$. Now consider the language $L := (\{a,b,d\}^* (ab^*c)^* \{a,b,d\}^*)^*$. So every string in L with a c also has an a preceding it, and no b may intervene between the two. It is easy to write an RLOC^1 formula that requires every node in the string to be labeled with exactly one symbol drawn from $\{a,b,c,d\}$. Thus it only remains to sufficiently restrict the distribution of c. If a is at most n steps to the left of c, this can be done by the formula

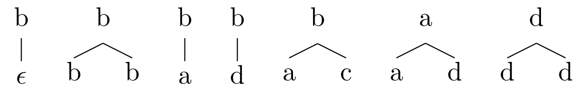
$$\phi := c \rightarrow \triangleright_1 a \vee (\triangleright_1 \triangleright_1 a \wedge \triangleright_1 d) \vee \ldots \vee (\triangleright_1^n a \wedge \triangleright_1 d \wedge \ldots \wedge \triangleright_1^{n-1} d),$$

where $\triangleright_1^n$ is a sequence of n many $\triangleright_1$. But by virtue of our formulas being required to be finite, the presence of an a can be enforced only up to n steps to the left of c. So if a is $n+1$ steps away from c, then ϕ will be false. Similar problems arise if one starts with a moving to the right. Nor is it possible to use d as an intermediary, as in, say, the formula $\psi := d \rightarrow (\triangleright_1 a \vee \triangleright_1 d) \wedge (\neg \triangleright_1 \triangleright_1 (a \vee \neg a) \rightarrow \triangleright_1 a)$, which forces every sequence of ds to be ultimately preceded by an a. The second conjunct is essential, since it rules out strings of the form $d^* c$. But ψ is too strong, because it is not satisfied by any L-strings containing the substrings bd or cd. Note that we cannot limit ψ to ds preceding c, again due to the finite length of RLOC^1 formulas, which is also the reason why we cannot write an implicational formula with b as its antecedent that will block bs from occurring between a and c. This

[1] Note that Φ can remain unchanged since the logical perspective allows for a node to be assigned multiple labels $l_1, \ldots, l_n$ instead of the sequence $\langle l_1, \ldots, l_n \rangle$ (which is the standard procedure in automata theory).

exhausts all possibilities, establishing that RLOC^1 fails to define L because it is in general incapable of restricting sequences of unbounded size.[2]

That L is definable in LOC^2 is witnessed by the following grammar of local 2-dimensional trees (with $\Sigma_0 := \{b\}$), which derives L without the use of additional features:



This case can be lifted to any dimension k by regarding L as the k-path language of some T^d, $1 \leq k \leq d$. □

Lemma 10 (MSO^d $\not\leq$ LOC^{d+1}). *There is a set Φ of* MSO^d *formulas, $d \geq 1$, such that there is no* LOC^{d+1} *definable set L^{d+1} whose d-dimensional yield is identical to* MOD(Φ).

Proof. Consider the language $L := (aa)^*$. Clearly, this language is definable in MSO^1 but not in first-order logic over strings, since it involves *modulo* counting. Hence it cannot be defined in LOC^1 either. We now show that LOC^2 is also too weak to define L. As $\Sigma := \{a\}$, the grammar for the tree language with L as its string yield can only consist of trees of depth 1 with all nodes labeled a. Clearly, none of the trees may have an odd number of leaf nodes, since this would allow us to derive a language with an odd number of as. So assume that all trees in our grammar have only an even number of leaves. But local tree sets are characterized by subtree substitution closure, whence we could rewrite a single leaf in a tree with an even number of leaves by another tree with an even numer of leaves, yielding a complex tree with an odd number of leaf nodes. This proves undefinability of L in LOC^2. We can again lift this example to any dimension $d \geq 2$ by viewing L as a path language. □

We now have LOC^k $<$ RLOC^k $<$ LOC^{k+1} and RLOC^k $<$ MSO^k and MSO^k $\not\leq$ LOC^{k+1} with respect to expressivity at dimension k, from which it follows immediately that a proper subset of all global constraints can be replaced by local ones.

Theorem 11 (Reducibility at lower dimensions). *Let C be the set of all k-global but not k-local constraints. Then C properly includes the set of all $c \in C$ for which there is a set Φ of* MSO^d *formulas, $k < d$, such that* MOD($\Phi \cup \{c\}$) *is recognizable and there is a $(k+1)$-local constraint c' with* MOD($\Phi \cup \{c\}$) = MOD($\Phi \cup \{c'\}$).

[2] A relaxed version of L is definable by an infinite set of RLOC^1 formulas. Let L' be the set of all strings over $\{a, b, c, d\}^*$ containing no substring of the form $(ad^*b^+d^*c)$ but a c does not have to be preceded by an a. Then one may write a formula ϕ that checks two intervals I, I' of size m and n, respectively. In particular, ϕ enforces that no b occurs in I if a is at the left edge of I and no c is contained in I and c is at the right edge of I' and no a is contained in I'. Occurrences of b in I' are banned in a symmetrical way. Pumping m and n independently gives rise to an infinite set of RLOC^1 formulas that defines L'.

Since RLOC^k is essentially a modal logic, we can even use model theoretic properties of modal logics, e.g. bisimulation invariance, to exhibit sufficient (but not necessary) conditions for reducibility of global constraints.

The results in this section have interesting implications for linguistic theorizing. The neutralizing effects of excessive feature coding with respect to NCCs lend support to recent proposals which try to do away with mechanisms of this kind in the analysis of phenomena such as pied-piping (e.g. [1]). That reducibility is limited to a proper subclass of the global constraints, on the other hand, provides us with a new perspective on approaches which severely constrain the size of representational locality domains by recourse to local constraints on derivations (e.g. [22]). In the light of my results, they seem to be less about reducing the size of locality domains — a quantitative notion — than determining the qualitative power of global constraints in syntax.

3 Comparative Constraints

3.1 Model Theory and Comparative Constraints — A Problem

Our interest in NCCs is almost entirely motivated by linguistic considerations. CCs, on the other hand, are intriguing from a mathematical perspective, too, because they make the well-formedness of a structure depend on the presence or absence of other structures, which is uncommon in model theoretic settings, to say the least. As we will see in a moment when we take a look at the properties of various subclasses of CCs, this peculiar trait forces us to move beyond a purely model theoretic approach, but — unexpectedly — not for all CCs.

According to Müller [21], CCs are either translocal or transderivational. Several years earlier, however, it had already been noticed by Potts [24] that the metarules of GPSG instantiate a well-behaved subclass of CCs. By definition, metarules are restrictions on the form of a grammar. They specify a template, and a grammar has to contain all rules that can be generated from said template. Metarules can be fruitfully applied towards several ends, e.g. to extract multiple constituent orders from a single rule or to ensure that the legitimacy of one construction in language L entails that another construction is licit in L, too. From these two examples it should already be clear that metarules, although they are stated as restrictions on grammars, serve in restricting entire languages rather than just the structures contained by them. In particular, metarules are a special case of closure conditions on tree languages. From this perspective, it is not too surprising that GPSG-style metarules are provably MSO^2-definable [24]. In fact, many closure constraints besides metarules can be expressed in MSO^2 as formula schemes [27]. With respect to the MS-hierarchy, this has several implications. First, there are more subclasses of CCs than predicted. Second, metarules instantiate a subclass that despite initial appearance can be represented by global constraints. Third, not all closure constraints are reducible to global constraints, since a formula scheme might give rise to an infinite set of formulas, which cannot be replaced by a single MSO formula of finite length.

Given that closure constraints are already more powerful than global constraints, the high position of translocal and transderivational constraints in the MS-hierarchy would be corroborated if closure constraints could be shown to be too weak to faithfully capture either class. This seems to be the case. Consider the translocal constraint *Avoid Pronoun*. Upon being handed a 2-dimensional tree $T \in L$, Avoid Pronoun computes T's reference set, which is the set of trees in L that can be obtained from T by replacing overt pronouns by covert ones and vice versa. Out of this set, it then picks the tree containing the fewest occurrences of overt pronouns as the optimal output candidate. One might try to formalize Avoid Pronoun as a closure constraint on L such that for every $T \in L$, no $T' \neq T$ in the reference set of T is contained in L. This will run into problems when there are several optimal output candidates, but it is only a minor complication compared to the greater, in fact insurmountable challenge a closure constraint implementation of Avoid Pronoun faces: it permits any output candidate T', not just optimal ones, to be in L as long as no candidates competing with T' belong to L. In other words, the closure constraint implementation of Avoid Pronoun allows for the selection of any candidate as the optimal output candidate under the proviso that all other output candidates are discarded. This means complete failure at capturing optimality, the very essence of CCs.

3.2 Comparative Constraints as Optimality Systems

From a model theoretic perspective, CCs are a conundrum. In order to verify that a set L of structures satisfies a CC, it does not suffice to look at L in isolation, we also have to consider what L looked like before the CC was applied to it. This kind of temporal reasoning is not readily available in model theory. Admittedly one could meddle with the models to encode such metadata, e.g. by moving to an ordered set of sets, but this is likely to obfuscate rather than illuminate our understanding of CCs in linguistics. Besides, trees are sets of nodes and tree languages are sets of sets of nodes, so our models would be sets of sets of sets of nodes and hence out of the reach of MSO, pushing us into the realm of higher-order logics and beyond decidability. For these reasons, then, it seems advisable to approach CCs from a different angle with tools that are already well-adapted to optimality and economy conditions: optimality systems (OSs) [5].

Definition 12 (Optimality system). *An* optimality system *over languages L, L' is a pair $\mathcal{O} := \langle \textsc{Gen}, C \rangle$ with $\textsc{Gen} \subseteq L \times L'$ and $C := \langle c_1, \dots, c_n \rangle$ a linearly ordered sequence of functions $c_i \colon \mathrm{range}(\textsc{Gen}) \to \mathbb{N}$. For $\langle i, o \rangle, \langle i, o' \rangle \in \textsc{Gen}$, $\langle i, o \rangle <_{\mathcal{O}} \langle i, o' \rangle$ iff there is an $1 \leq k \leq n$ such that $c_k(o) < c_k(o')$ and for all $j < k$, $c_j(o) = c_j(o')$. The* output language of $\mathcal{O}$ *is $L_{\mathcal{O}} := \mathrm{range}(\{\langle i, o \rangle \in \textsc{Gen} \mid$ there is no o' such that $\langle i, o' \rangle <_{\mathcal{O}} \langle i, o \rangle\})$.*

The idea underlying OSs is very simple and taken directly from Optimality Theory [25]. Given some input language L, we compute for every $i \in L$ its set of output candidates, i.e. the set of $o \in L'$ such that $\langle i, o \rangle \in \textsc{Gen}$. We then determine for each such o how often it violates the constraint c_1 and only the candidates with the fewest violations are kept as possible output candidates. All other candidates are discarded. We then proceed analogously for c_2 until c_n. The remaining output candidates are the optimal output candidates for i. In sum, an OS filters the set of output candidates in a stepwise manner while ensuring all along the way that the set is never emptied.

It should be easy to see that without further restrictions, any recursively enumerable language can be derived by an OS. Interestingly, though, it has been established in a series of papers [5, 7, 11, 13, 37] that an OS defines a rational transduction if all the conditions below are satisfied (the converse does not hold).[3]

- L is a recognizable set.
- Gen is a rational relation.
- Every constraint defines a rational relation on the set of competing output candidates.
- Optimality is global: If $o \in L'$ is an optimal output candidate for $i \in L$, then there is no $i' \in L$ such that o is an output candidate for i' but not an optimal one (see [11] for details).

It is a well-known fact that recognizable sets are closed under rational transductions. Therefore, if an OS is equivalent to a rational transduction, then its output language is recognizable. Recall that recognizability entails definability in MSO, so the output language of such an OS has to be definable in terms of global constraints. Equating CCs with the subclass of OSs where $\text{Gen} \subseteq L \times L$, this yields an intriguing (albeit partial) characterization of reducibility for CCs. In the following, we let $\mathcal{O}(\Phi)$ denote the output language of $\mathcal{O}$ with $L = \text{Mod}(\Phi)$.

Theorem 13. *Let Φ be a set of* MSO^k *formulas and c_c a comparative constraint obeying the conditions listed above. Then there is an* MSO^k *formula c_g such that $\mathcal{O}(\Phi) = \text{Mod}(\Phi \cup \{c_g\})$.*

Proof. We know from our previous discussion that $L_{\mathcal{O}} = \mathcal{O}(\Phi)$ is recognizable. Since recognizable sets are closed under complement, there is an MSO^k formula ϕ with $\text{Mod}(\phi) = \text{Mod}(\Phi) \setminus \mathcal{O}(\Phi)$. Then $c_g := \neg\phi$. □

In other words, a CC of this restricted type can be reduced to a global constraint.

On a methodological level, Theorem 13 allows linguists to freely employ a subclass of all CCs without running danger of computational intractability — a lot of the criticism commonly leveled against translocal and transderivational accounts [10, 34] should thus turn out to be unfounded.[4] The astute reader may

[3] Kepser and Mönnich [15] define similar conditions for cases where L is a linear context-free language. As a consequence, the results in this section apply just as well to conservative extensions of the recognizable languages of d-dimensional trees, for instance Minimalist languages.

[4] On a speculative note, one may interpret Theorem 13 on an ontological level, that is as a claim that CCs are prevalent in the early stages of language acquisition but are subsequently recompiled into NCCs. This could offer a new perspective on well-known phenomena such as Principle B delay [9].

wonder, though, how many CCs from the linguistic literature are to be found in said subclass. This is a valid concern. While I do not have any conclusive answers yet, it seems that most syntactic (in contrast to most semantic) CCs satisfy global optimality [8]. Hence their reducibility hinges solely on their reference sets and their economy metric being rational relations, which I expect to hold for many interesting cases.

Conclusion

I demonstrated that the intuitive constraint hierarchy of [21, 23] can be given a rigorous foundation that mostly confirms the big picture envisioned by these authors (with the addition of closure constraints as a third macro-class, inbetween non-comparative and comparative constraints). The interesting catch, however, is that certain constraints can be reduced to simpler ones depending on parameters such as the feature signature and the dimensionality and branching factor of our structures. From an application perspective, the reducibility of non-comparative constraints is of limited interest, due to the power of alternative feature coding techniques; the reducibility of comparative constraints, on the other hand, has profound repercussions as it opens up a pathway to their efficient implementation in natural language processing systems. Both types of reducibility results are of eminent importance to linguistic issues, foremost the study of locality phenomena and the Minimalist dictum that language is an optimal cognitive device.

Acknowledgments. I am grateful to Ed Stabler and Sarah Zobel for helpful discussion and their extensive comments on numerous inferior incarnations of this paper.

References

[1] Cable, S.: The Grammar of Q: Q-Particles and the Nature of Wh-Fronting. Ph.D. thesis, MIT (2007)
[2] Chomsky, N.: A minimalist program for linguistic theory. In: Hale, K., Keyser, S.J. (eds.) The View from Building 20, pp. 1–52. MIT Press, Cambridge, Mass. (1993)
[3] Chomsky, N.: The Minimalist Program. MIT Press, Cambridge, Mass. (1995)
[4] Chomsky, N., Schützenberger, M.P.: The algebraic theory of context-free languages. In: Braffort, P., Hirschberg, D. (eds.) Computer Programming and Formal Systems, pp. 118–161. Studies in Logic and the Foundations of Mathematics, North-Holland, Amsterdam (1963)
[5] Frank, R., Satta, G.: Optimality theory and the generative complexity of constraint violability. Computational Linguistics 24, 307–315 (1998)
[6] Gazdar, G., Klein, E., Pullum, G.K., Sag, I.A.: Generalized Phrase Structure Grammar. Blackwell, Oxford (1985)

[7] Gerdemann, D., van Noord, G.: Approximation and exactness in finite-state phonology. In: Eisner, J., Karttunen, L., Thériault, A. (eds.) Finite State Phonology. Proceedings SIGPHON 2000, ACL (2000)

[8] Graf, T.: Reference sets and congruences (in progress), ms., University of California, Los Angeles

[9] Grodzinsky, Y., Reinhart, T.: The innateness of binding and coreference. Linguistic Inquiry 24, 69–102 (1993)

[10] Johnson, D., Lappin, S.: Local Constraints vs. Economy. CSLI, Stanford (1999)

[11] Jäger, G.: Gradient constraint in finite state OT: The unidirectional and the bidirectional case. In: Kaufmann, I., Stiebels, B. (eds.) More than Words. A Festschrift for Dieter Wunderlich, pp. 299–325. Akademie Verlag, Berlin (2002)

[12] Kanazawa, M., Salvati, S.: Generating control languages with abstract categorial grammars. In: Kallmeyer, L., Monachesi, P., Penn, G. (eds.) Formal Grammar FG 2007. CSLI Publications (2007), `http://www.labri.fr/publications/mef/2007/KS07`

[13] Karttunen, L.: The proper treatment of optimality in computational phonology (1998), manuscript, Xerox Research Center Europe

[14] Kasper, R., Kiefer, B., Netter, K., Vijay-Shanker, K.: Compilation of HPSG to TAG. In: Proceedings of the 33rd annual meeting of the Association for Computational Linguistics. pp. 92–99 (1995)

[15] Kepser, S., Mönnich, U.: Closure properties of linear context-free tree languages with an application to optimality theory. Theoretical Computer Science 354, 82–97 (2006)

[16] Kobele, G.M., Michaelis, J.: Two type-0 variants of minimalist grammars. In: FG-MoL 2005. The 10th conference on Formal Grammar and the 9th Meeting on Mathematics of Language. pp. 81–93. Edinburgh (2005)

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

[18] Kracht, M.: Syntactic codes and grammar refinement. Journal of Logic, Language and Information 4, 41–60 (1995)

[19] Kracht, M.: Inessential features. In: Lecomte, A., Lamarche, F., Perrier, G. (eds.) Logical Aspects of Computational Linguistics. Springer, Berlin (1997)

[20] Michaelis, J.: Derivational minimalism is mildly context-sensitive. Lecture Notes in Artificial Intelligence 2014, 179–198 (1998)

[21] Müller, G.: Constraints in syntax. Lecture Notes, Universität Leipzig (2005)

[22] Müller, G.: On deriving CED effects from the PIC (2008), ms., Universität Leipzig. To appear in Linguistic Inquiry 41:1 (2010).

[23] Müller, G., Sternefeld, W.: The rise of competition in syntax: A synopsis. In: Sternefeld, W., Müller, G. (eds.) Competition in Syntax, pp. 1–68. Mouton de Gruyter, Berlin (2000)

[24] Potts, C.: Three kinds of transderivational constraints. In: Mac Bhloscaidh, S. (ed.) Syntax at Santa Cruz, vol. 3, pp. 21–40. Linguistics Department, UC Santa Cruz, Santa Cruz (2001)

[25] Prince, A., Smolensky, P.: Optimality Theory: Constraint Interaction in Generative Grammar. Blackwell, Oxford (2004)

[26] Reinhart, T.: Anaphora and Semantic Interpretation. Croon-Helm, Chicago University Press (1983)

[27] Rogers, J.: "Grammarless" phrase structure grammar. Linguistics and Philosophy 20, 721–746 (1997)

[28] Rogers, J.: Strict LT_2 : Regular :: Local : Recognizable. In: Retoré, C. (ed.) Logical Aspects of Computational Linguistics: First International Conference, LACL '96 (Selected Papers). Lectures Notes in Computer Science/Lectures Notes in Artificial Intelligence, vol. 1328, pp. 366–385. Springer (1997)

[29] Rogers, J.: A Descriptive Approach to Language-Theoretic Complexity. CSLI, Stanford (1998)

[30] Rogers, J.: A descriptive characterization of tree-adjoining languages. In: Proceedings of the 17th International Conference on Computational Linguistics (COLING'98) and the 36th Annual Meeting of the Association for Computational Linguistics (ACL'98). pp. 1117–1121 (1998)

[31] Rogers, J.: Syntactic structures as multi-dimensional trees. Research on Language and Computation 1(1), 265–305 (2003)

[32] Rogers, J.: wMSO theories as grammar formalisms. Theoretical Computer Science 293, 291–320 (2003)

[33] Stabler, E.P.: Derivational minimalism. In: Retoré, C. (ed.) Logical Aspects of Computational Linguistics: First International Conference, LACL '96, Nancy, France, September 23-25, 1996. Selected Papers, pp. 68–95. Springer, Berlin (1997)

[34] Stroik, T.S.: Locality in Minimalist Syntax. MIT Press, Cambridge, Mass. (2009)

[35] Takahashi, S., Fox, D.: MaxElide and the re-binding problem. In: Georgala, E., Howell, J. (eds.) Proceedings of SALT XV. pp. 223–240. CLC publications, Ithaca, NY (2005)

[36] Thatcher, J.W.: Characterizing derivation trees for context-free grammars through a generalization of finite automata theory. Journal of Computer and System Sciences 1, 317–322 (1967)

[37] Wartena, C.: A note on the complexity of optimality systems. In: Blutner, R., Jäger, G. (eds.) Studies in Optimality Theory, pp. 64–72. University of Potsdam, Potsdam, Germany (2000)