Relative clauses as a benchmark for Minimalist parsing

Thomas Graf, James Monette, and Chong Zhang
Department of Linguistics, Stony Brook University, USA

ABSTRACT

Minimalist grammars have been used recently in a series of papers to explain well-known contrasts in human sentence processing in terms of subtle structural differences. These proposals combine a top-down parser with complexity metrics that relate parsing difficulty to memory usage. So far, though, there has been no large-scale exploration of the space of viable metrics. Building on this earlier work, we compare the ability of 1,600 metrics to derive several processing effects observed with relative clauses, many of which have been proven difficult to unify. We show that among those 1,600 candidates, a few metrics (and only a few) can provide a unified account of all these contrasts. This is a welcome result for two reasons: First, it provides a novel account of extensively studied psycholinguistic data. Second, it significantly limits the number of viable metrics that may be applied to other phenomena, thus reducing theoretical indeterminacy.

Keywords: Parsing, sentence processing, Minimalist grammars, memory usage, relative clauses, promotion analysis, English, East Asian

1 INTRODUCTION

It is beyond doubt that the structural properties of a sentence influence how easily said sentence is processed by humans. For example, a sentence with multiple levels of center embedding is harder to parse than its counterpart with right embedding, and English subject relative clauses are processed more quickly than object relative clauses. There is large disagreement, however, on what the relevant structural properties are. This paper continues a recent series of investigations (Kobele *et al.* 2013; Graf and Marcinek 2014; Graf *et al.* 2015; Gerth

2015) that approach this question by combining Stabler's (2013) top-down parser for Minimalist grammars (MGs) with structurally rich analyses from Minimalist syntax, the most recent version of Chomsky's transformational grammar framework.

The works above are part of longer tradition applying computational formalisms to human sentence processing (Joshi 1990; Rambow and Joshi 1995; Steedman 2001; Hale 2011; Yun *et al.* 2014, among others). Common to all of them is a tripartite structure consisting of I) an articulated theory of syntax that has sufficient empirical coverage to be applicable to a wide range of constructions, II) a sound and complete parser for the syntactic formalism, and III) a complexity metric that acts as linking theory to derive psycholinguistic predictions from the previous two components. The allure of this approach is that all components are rigorously specified and mathematically worked out to a degree that allows for very fine-grained and detailed processing predictions. Not only does this provide insightful explanation of certain processing phenomena, it also makes it possible to distinguish between competing syntactic proposals based on their psycholinguistic predictions.

The decomposition into three distinct modules is intuitive and elegant, but it also highlights a worrying underspecification issue. With three components that by necessity have to make very detailed assumptions, it is to be expected that a large number of different combinations all replicate the same behavioral data. For instance, a syntactician whose analysis makes the wrong processing predictions may insist that the problem is not with the analysis but with the parser or the complexity metric. For the previous MG modeling work, the issue has already arisen with respect to the choice of complexity metrics, with Graf and Marcinek (2014) and Graf *et al.* (2015) arguing for slightly different metrics than Kobele *et al.* (2013) and Gerth (2015). The specificity of computational models – one of their biggest virtues – thus runs the risk of combinatorial explosion and empirical indeterminacy, which would severely weaken their appeal.

In order to address this issue, we define a total of 1,600 complexity metrics and evaluate whether they can account for the processing contrasts with relative clauses that were originally discussed in Kobele *et al.* (2013), Graf and Marcinek (2014), and Graf *et al.* (2016). We also use two different analyses of relative clauses from Minimal-

ist syntax (promotion and wh) to determine whether the set of empirically viable metrics is still sufficiently structure-sensitive to distinguish between the accounts. Our findings show that the issues of indeterminacy and combinatorial explosion are much less severe in practice than one might expect – a handful of data points is sufficient to significantly reduce the space of empirically viable parsing models. Furthermore, this reduced set contains some very simple metrics that are capable of explaining a wide range of processing contrasts in an intuitively pleasing fashion. At the same time, the set of viable metrics varies with the posited analysis in an interesting way, which suggests that more data points will eventually allow us to rule out specific syntactic proposals.

The paper proceeds as follows. The next section introduces MGs (2.1) and explains how the behavior of Stabler's (2013) top-down parser for MGs can be represented at an abstract level with index/outdex annotated derivation trees. Section 3 then defines 1,600 complexity metrics that operate over these annotated derivation trees. This large number is obtained from just three basic metrics that are subsequently parameterized along various axes. In Section 4, we establish the empirical viability of only a few of these 1,600 metrics for the processing of relative clauses in English, Chinese, Korean, and Japanese. The paper concludes with a discussion of conceptual and methodological aspects of our finding.

2 MINIMALIST GRAMMARS FOR PROCESSING

The mathematical backbone of this paper is provided by MGs (Stabler 1997, 2011) and the top-down MG parser proposed by Stabler (2013). MGs were chosen because they present a rare combination of traits. On the one hand they incorporate ideas from Chomskyan syntax that have found wide adoption in syntactic processing. MGs are also flexible enough to implement even unusual proposals such as Late Merge (Lebeaux 1988; Takahashi and Hulsey 2009) and test their predictions. On the other hand they can be regarded as a simple variant of context-free grammars, which have been studied extensively in the computational parsing literature (Shieber *et al.* 1995; Sikkel 1997). MGs thus act as mathematical glue between formal parsing theory, psycholinguistics, and large areas of contemporary syntax.

After a brief introduction to MGs in Section 2.1, we discuss the central role of derivation trees (2.2) and how Kobele *et al.*'s (2013) system of annotating derivation trees acts as a high-level abstraction of Stabler's top-down parser (2.3). This provides us with a unified representational format that simultaneously describes the structure of a sentence and relevant parts of its parse history. When combined with the complexity metrics in Section 3, this simple system is sufficient to obtain concrete processing predictions.

2.1 *Non-technical introduction to Minimalist grammars*

MGs take inspiration from the most recent iteration of transformational grammar, known as *Minimalism* or *Minimalist syntax* (Chomsky 1995b, 2001). Like all iterations of transformational grammar, MGs furnish a mechanism for encoding basic head-argument relations which are then manipulated by a movement operation to produce the actual syntactic structure. In the case of MGs, these two modes of structure building are called *Merge* and *Move*, respectively. What differentiates MGs from standard Minimalism is their fully explicit feature calculus, which regulates when each operation has to be applied. This makes MGs a lexicalized formalism similar to CCG, LFG or HPSG in the sense that each grammar G is just a finite list of feature-annotated lexical items (LIs) – a structure is generated by G iff it can be assembled from LIs of G according to their feature specifications.

For the purposes of this paper, an intuitive understanding of MGs is sufficient, so we do not give a complete, rigorous definition here. The interested reader is referred to Stabler (2011), Graf (2013, Chapters 1 & 2), and Gerth (2015, Section 4.1) for detailed yet accessible introductions. Suppose that we want to generate the sentence *John, the girl likes*. While there are in principle infinitely many distinct ways to do so with MGs, only a few, marginally different ones are also entertained in Minimalist syntax. The most common analysis posits the syntactic structure shown in Figure 1, where dashed lines have been added to indicate which positions certain phrases were moved from. The basic idea is that the sentence starts out as *the girl likes John* and is subsequently transformed into *John, the girl likes* via Move. There are, however, several independently motivated factors that complicate this simple picture.

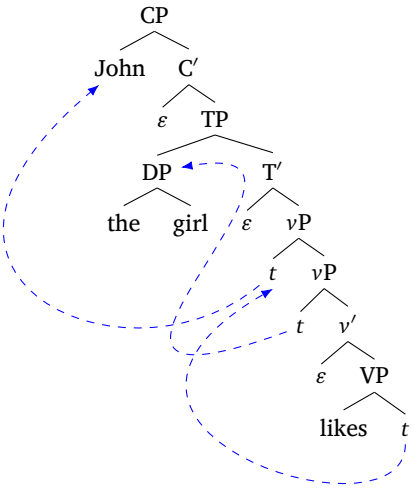


Figure 1:
MG phrase structure tree for *John, the girl likes*; dashed arrows indicate movement

- Instead of the usual X′-structure, a more succinct Bare Phrase Structure tree (Chomsky 1995a) is assumed. The two are almost identical except that Bare Phrase Structure trees omit all unary branches (wherefore they are strictly binary branching). So an X′-structure like $[_{\text{DP}}\ [_{\text{D}'}\ [_{\text{D}}\ \textit{John}\ ]]]$ reduces to simply *John*.
- A phrase can have multiple specifiers but only one complement. Heads are always linearized to the left of their complement and to the right of their specifiers (Kayne 1994).
- Sentences are allowed to contain unpronounced LIs, which are denoted ε.
- VPs are split into VP and *v*P (read "little VP") following ideas first formulated in Larson (1988). The *v*P phrase serves many purposes in the literature, but its relevance for this paper is limited to its role as the base position for subjects.
- Subjects in English (and all other languages discussed in this paper) move from the lowest specifier of *v*P to the canonical subject position, i.e. the lowest specifier of TP.
- For the sake of exposition, Figure 1 also incorporates the assumption that a phrase moving out of *v*P must intermittently land in a specifier of *v*P (Chomsky 2001). We omit this in the remainder of the paper as it has no effect on our results and thus would only add irrelevant complexity.

While many of these assumptions are not widely shared outside of Minimalist syntax and add a certain degree of verbosity, they rest on decades of syntactic research. Since a major goal of the MG parsing project is to determine to what extent different syntactic assumptions can affect processing predictions, we adapt these Minimalist analyses as faithfully as possible to MGs.

Let us then look more closely at how the phrase structure tree in Figure 1 is assembled by an MG. MGs combine LIs into trees via the structure-building operations *Merge* and *Move* based on the features carried by those LIs. We start out by applying Merge to *likes* and *John*, which marks *John* as an argument of *likes*. In order for this Merge step to be licensed, *likes* must have a feature that indicates that the verb takes a DP as its complement, whereas *John* must have the matching category feature D. The verb is then selected by the unpronounced head of *v*P, which also requires a DP as its specifier. In this particular case, the DP is obtained by merging *the* with *girl*. As before, all these instances of Merge must be licensed by suitable feature specifications on the LIs. We do not write out these features here as they will play no role in this paper beyond the fact that an MG parser needs to keep track of all features.

At this point we have assembled the tree depicted in Figure 2. This figure also shows the corresponding *derivation tree*, which records the structure-building steps taken to build the phrase structure tree. In a derivation tree, all leaves are lexical items, and all interior nodes are labeled by Merge and Move depending on which operation takes place at that point. The two trees in Figure 2 differ only in their interior node labels, but they will diverge more significantly once Move enters the picture.

So far each step has added new material to the tree via Merge. But now something different happens: the object DP *John* is displaced to a specifier of *v*P via Move. When exactly Move may take place and

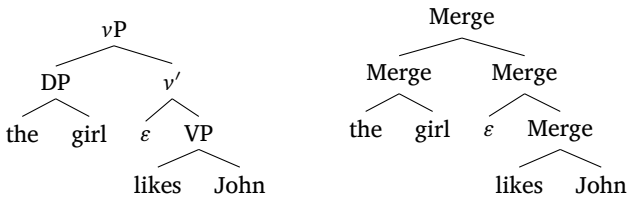


Figure 2: An intermediate state of the assembly of *John, the girl likes*; all feature specifications are omitted

which phrase it may displace is once again controlled by the feature calculus. In the case at hand, the *v*P-head must have a feature f that requires some phrase to move into a *v*P specifier. Similarly, *John* must have a feature that requires it to undergo f-movement. The result of Move is shown in Figure 3. Note that the phrase structure tree and the derivation tree now have different geometries, with *John* in a *v*P specifier in the former but still in its base position in the latter. Consequently, reading the leaves in a derivation tree from left to right thus may not produce the actual word order of the sentence, which will play an important role during the discussion of MG parsing in the next two sections.

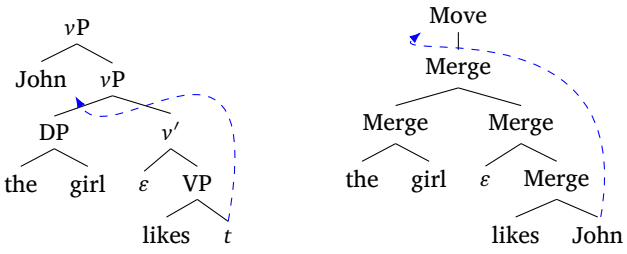


Figure 3:
Phrase structure tree and derivation tree after the first movement step; dashed arrows are not part of the trees

The reader may also wonder why Move is represented as a unary branching node even though the operation seems to involve two arguments, a target position and the subtree that is to be displaced. The answer is that Move is a deterministic operation in MGs. The target position is always added at the root of the current tree, and from the feature specifications of LIs one can always infer which particular subtree is to be displaced. There simply is no need to explicitly specify the arguments of Move. However, in the derivation trees in this paper we omit the feature specifications for the sake of brevity and instead use dashed arrows to indicate what moves where.

Strictly speaking we could stop here as the current phrase structure tree already has *John, the girl likes* as its string yield. However, the tree is still incomplete according to Minimalist syntax and thus the derivation continues. After merging the tree with an unpronounced T-head, the subject *the girl* moves from its base position inside *v*P to the canonical subject position in the specifier of TP. Then the TP is merged with an unpronounced C-head, and *John* is topicalized by moving it to a CP specifier. By assumption, a tree is well-formed iff its root is

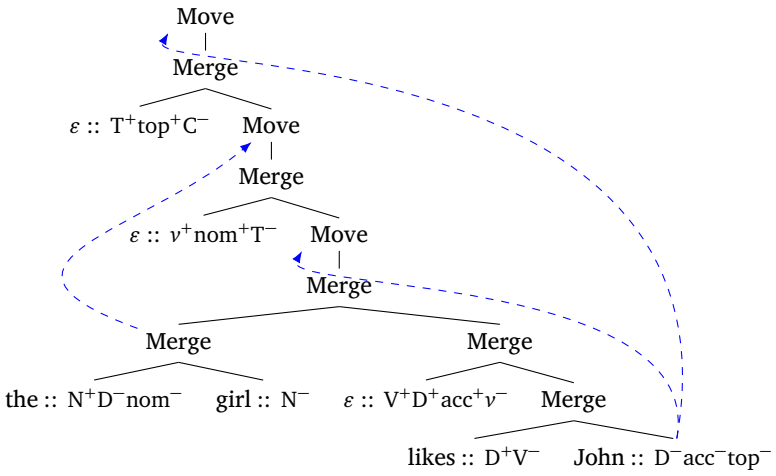


Figure 4: MG derivation tree for *John, the girl likes* with explicit feature specifications for all LIs

labeled CP and all feature requirements have been satisfied. Since this is the case for this tree, the derivation can stop here. The full derivation is given in Figure 4 – to give the reader an idea of what the MG feature calculus looks like, we also list the feature specifications in this example.

2.2 *The central role of derivation trees*

Since derivation trees provide a record of how a given phrase structure tree is to be assembled, they implicitly contain all the information encoded in the latter. In itself this is a rather unremarkable fact, but in the MG community a trend has developed in the last 10 years to treat derivation trees as the primary data structure of MGs (Kobele *et al.* 2007; Kobele 2011, 2015; Graf 2011, 2012a,b, 2013; Hunter 2011). That is to say, MGs are no longer viewed as generators of phrase structure trees or strings but rather as a generator of derivation trees. A suitable graph transduction then transforms the derivation trees into the desired output structure – strings, phrase structure trees, logical forms, dependency graphs, and so on. Similar ideas have been explored in a more general setting under the label of two-step approaches (Morawietz 2003; Mönnich 2006), interpreted regular tree grammars (Koller and Kuhlmann 2011), and Abstract Categorial Grammar (de Groote 2001). This view of MGs has many technical advantages, but it also provides a unique perspective on parsing: if one assumes that the structures to be produced by an MG parser are derivation trees rather than

phrase structure trees, MG parsing turns out to be closely related to parsing of context-free grammars (CFGs).

It has been known for a while now that an MG's set of well-formed derivation trees forms a regular tree language (Michaelis 2001; Kobele *et al.* 2007; Salvati 2011; Graf 2012a).[1] Regular tree languages, in turn, can be directly linked to CFGs. For any CFG G, let $D(G)$ be the set of its derivation trees. Furthermore, a projection $\pi : \Sigma \to \Omega$ is a total function from alphabet Σ to alphabet Ω. Projections can be extended to trees in a point-wise fashion such that the image of tree t under π is the result of replacing each label in t by its image under π. A famous theorem by Thatcher (1967) states that a tree language L is regular iff there is a CFG G and projection π such that $L = \pi(D(G))$. In other words, every regular tree language can be generated by a CFG if one is willing to refine the node labels. For MGs, the refinement involves replacing all instances of Merge and Move with tuples of feature specifications. The details are of no particular interest here (see Michaelis 2001, Kobele *et al.* 2007 and section 2.1.1 of Graf 2013). The crucial point is that MGs have a close link to CFGs via their derivation trees, and this link can be exploited in parsing.

An MG parser can co-opt CFG parsing techniques as long as it has mechanisms to deal with the properties that separate MGs from CFGs. The use of Merge and Move as interior node labels instead of more fine-grained labels is rather trivial in this respect. The true challenge lies in the fact that the left-to-right order of leaves in the MG derivation tree does not correspond to the linear order in the string. The latter can be deduced from the former only if one keeps track of the structural alterations brought about by Move, which requires some ingenuity. At any rate, MGs can be regarded as CFGs with a more complex mapping

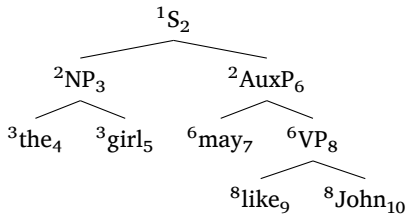
[1] This fact is not restricted to standard MGs as presented in the previous section. MGs have been extended and modified with numerous devices from the syntacticians' toolbox. Even the most truncated list includes adjunction (Frey and Gärtner 2002; Fowlie 2013; Hunter 2015a), new movement types (Kobele 2006; Stabler 2006; Gärtner and Michaelis 2010; Graf 2012b), and a variety of constraints (Gärtner and Michaelis 2007; Graf 2013). But these revised versions still preserve the regularity of the derivation tree language. The complexity of the string yield mapping is affected by new movement types, but stays within the restricted class of transductions that are definable in first-order logic with equality and proper dominance (Graf 2012b).

from trees to strings, and MG parsers are CFG parsers that have been augmented with a mechanism to handle this increased complexity.[2]

2.3 *Encoding parses with tree annotations*

Consider a standard recursive descent parser for CFGs, i.e. a parser that operates top-down, depth-first, and left-to-right. Following Kobele *et al.* (2013), the order in which a parser builds a given tree t for input string i can be represented by a specific annotation of the nodes of t as in Figure 5.

Figure 5:
The annotations of the tree indicate in what order it is built by a recursive descent parser



Intuitively, the annotation indicates for each node in the tree when it is first conjectured by the parser and at what point it is considered completed and flushed from memory. So at the very first step, the parser conjectures S, which is expanded in step 2 to NP and AuxP. Assuming that NP and AuxP will eventually yield the desired string, S can be marked as done and removed from memory at step 2. After that the parser works on NP (because it operates left-to-right), adding *the* and *girl*. So those two are first conjectured at step 3 while NP is removed from memory at the same time. As the parser is depth-first, it now proceeds to work on *the* and *girl* rather than AuxP. First *the* is scanned at step 4. This means that the parser reads in the first word of

[2] The connection between MGs and CFGs does not emerge with phrase structure trees. MGs are known to be weakly equivalent to MCFGs (Harkema 2001; Michaelis 2001), from which it follows immediately that there are MGs whose set of well-formed phrase structure trees is supra-regular. But supra-regular tree languages cannot be made context-free via a simple relabeling as is the case for regular tree languages, wherefore CFG parsing techniques are not easily extended to Minimalist phrase structure trees. The technical gulf between phrase structure trees on the one hand and derivation trees on the other is significant, and it holds only because derivation trees need not directly encode the linear order of leaves in the string.

the input and verifies that it is indeed *the*. If so, the parser then scans *girl* at step 5, checking it against the second word in the string. Assuming that scanning succeeded, the parser then returns to AuxP, which it has held in memory since step 2. Now at step 6 it finally gets to flush AuxP from memory and replace it by *may* and VP. The remainder of the parse is straight-forward.

This is of course a highly abstracted view of the actual work done by a parser. For one thing, a parser operates with parse items rather than trees or tree nodes, and how such parse items are organized in memory depends on a lot on the specifics of the algorithm (for example, a chart-based parser would never remove an already constructed parse item from memory). More importantly, the problem of which rewrite rules must be chosen to derive the correct string is completely ignored. So this way of annotating trees is no substitute for a proper, rigorously defined parser. Crucially, though, these abstractions are immaterial for this paper's approach to modeling human sentence processing – the tree annotation is a sufficiently close representation of a parser's behavior to enable the kind of processing predictions we are interested in.

Now consider how a standard recursive descent parser would operate over an MG derivation tree. Consider first the derivation tree for *the girl likes John*, depicted in Figure 6. For the sake of clarity, we indicate unpronounced LIs by their category (C, T, *v*). As can be gleaned from the figure, the parser scans the leaf nodes in this derivation in the following order: C T *the girl v likes John*. But the actual order in the input string is C *the girl* T *v likes John*, with *the girl* preceding T rather than following it. In this particular case the slight difference does not matter because T is the empty string, so "T *the girl*" and "*the girl* T" are exactly the same string. However, this example already shows that a standard recursive descent parser is not guaranteed to scan the leaf nodes of an MG derivation in the order in which they appear in the input string. Problems arise whenever Move actually alters the precedence relations between leaf nodes, as is the case with *John, the girl likes* (also shown in Figure 6). The recursive descent parser will reach *the* and try to scan it. It subsequently aborts the parse because the scanned leaf node does not match the first word in the input, *John*. We see, then, that a CFG recursive descent parser does not operate correctly over MG derivation trees despite them being context-free.

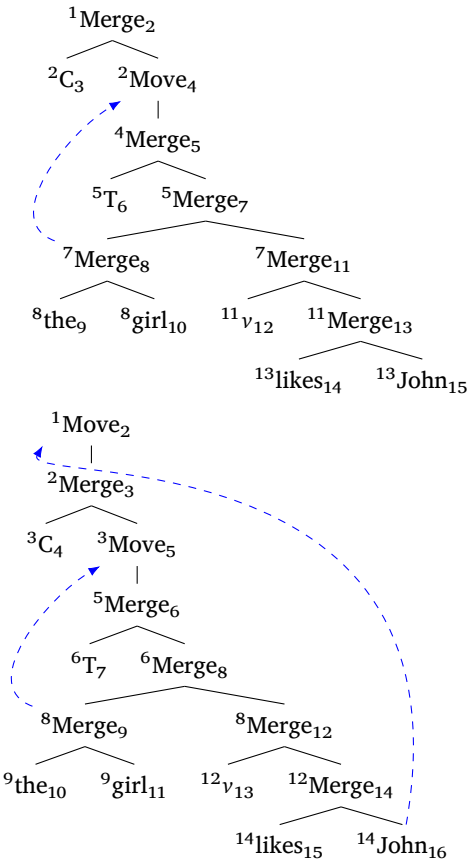


Figure 6:
Standard recursive descent works for MG derivation trees only if Move does not alter the precedence relations in the string

The problem with the CFG recursive descent parser is its assumption that the left-to-right order in trees reflects the left-to-right order in the derived string. The core insight of Stabler (2013) (building on Mainguy 2010) is that the left-to-right order can instead be inferred from the MG feature calculus. At the level of abstraction used in this paper, the answer is even simpler. Given two sibling nodes m and n in an MG derivation, m is left of n iff m reflexively dominates a leaf node l such that every leaf node reflexively dominated by n is somewhere to the right of l in the derived string (where reflexive dominance is the reflexive, transitive closure of the mother-of relation). According to this definition, the recursive descent parser will choose a right branch instead of a left one whenever the right branch contains a mover and this mover appears to the left of all the material in the left branch.

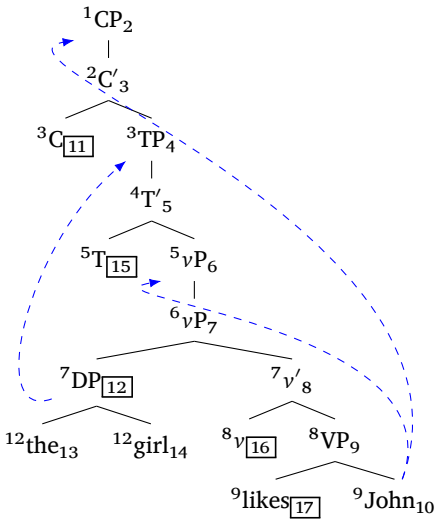


Figure 7:
Modified recursive descent operates correctly over MG derivation trees

Figure 7 shows that this modified kind of recursive descent scans the leaf nodes in the correct order: *John* C *the girl* T *v* likes. To further increase the readability of derivation trees, this and all later figures replace the labels Merge and Move by the corresponding X′-labels in the phrase structure tree.

The annotations for MG derivation trees can be computed in a purely tree-geometric fashion (Graf *et al.* 2015). From here on out, we will refer to a node's superscript as its *index* and its subscript as its *outdex*. The terminology is intended to highlight that the index represents the step at which the parser first conjectures a node whereas the outdex records the point at which it has finished working on the node. Index and outdex thus provide information about the parser's memory usage. The greater the difference between the two, the longer an item has to be stored in memory. Since memory usage plays a central role in deriving processing predictions from these annotations, any outdex that is larger than the corresponding index by a non-trivial amount will be henceforth highlighted by a box (more on this in Section 3.2).

Definition 1 *Let s[urface]-precedence be the relation that holds between nodes m_d and n_d in a derivation tree iff their counterparts m_p and n_p in the corresponding phrase structure tree stand in the precedence relation. If m_d undergoes movement during the derivation, its counterpart m_p is the final landing site rather than its base position.*

Given a well-formed Minimalist derivation tree t, its index/outdex annotation is computed as follows:

1. *Every node of t has exactly one index and exactly one outdex.*
2. *The index of the root is 1. For every other node, its index is identical to the outdex of its mother.*
3. *If nodes n and n' are distinct nodes with index i, and n reflexively dominates a node that is not s-preceded by any node reflexively dominated by n', then n has outdex $i + 1$.*
4. *Otherwise, the outdex of node n with index i is $\max(i+1, j+1)$, where $j \geq 0$ is greatest among the outdices of all nodes that s-precede n but are not reflexively dominated by n.*

Throughout the rest of the paper we use these annotated derivation trees as abstract representations of the behavior of Stabler's (2013) recursive descent parser for MGs. This greatly simplifies the discussion by substituting easily interpreted derivation trees with indices and outdices for the complex mechanics of the parser. But it means that the difficulty of finding this derivation tree in the first place is completely ignored. The most demanding task of parsing – searching through a large space of structures in the search for the correct one – is taken out of the equation. This simplification is shared among all recent work that use Stabler's MG parser to model human processing (Kobele *et al.* 2013; Graf and Marcinek 2014; Graf *et al.* 2015; Gerth 2015). In the words of Graf *et al.* (2015, p.3):

> While psychologically implausible, this idealization is meant to stake out a specific research goal: processing effects must be explained purely in terms of the syntactic complexity of the involved structures, rather than the difficulty of finding these structures in a large space of alternatives. More pointedly, we assume that parsing difficulty *modulo* nondeterminism is sufficient to account for the processing phenomena under discussion.

The aim of these MG processing models, then, is to see how much of human sentence processing can be explained by considering only the order of how the parts of the correct derivation are built. This does not deny that ambiguity has a large role to play, e.g. in garden path

sentences, but it is taken out of the equation in order to determine the relevance of isolated structural factors. A simpler model has the advantage of being easier to reason about, and the focus on structure allows us to compare specific syntactic proposals according to their processing predictions.

3 COMPLEXITY METRICS FOR PROCESSING

The previous section recast Stabler's top-down parsing algorithm for MGs as a particular kind of tree annotation, but this raises the question how a simple annotation of derivation trees can be linked to psycholinguistic processing effects. This is accomplished via a linking theory, which takes the form of *complexity metrics*.[3] The next section discusses what we mean by complexity metrics and how all our metrics are rooted in notions of memory usage. Sections 3.2 and 3.3 then provide formal definitions of all the relevant metrics. The full set comprises 1,600 metrics, of which only a handful will prove able to account for all the data in Section 4.

3.1 *Complexity metrics and three notions of memory usage*

A complexity metric is any procedure that ranks strings according to processing difficulty. For instance, Kimball's (1973) principle that the human parser cannot work on more than two CPs at the same time provides a simple complexity metric that is computed over phrase structure trees. O'Grady (2011) suggests that the length of movement dependencies affects processing difficulty. The Derivational Theory of Complexity (Miller and Chomsky 1963; Miller and McKean 1964; see also Phillips 1996, Chapter 5) equates complexity with the number of syntactic operations that are required to build said sentence. Syntactic Prediction Locality Theory (Gibson 1998) and Dependency Locality Theory (Gibson 2000) instead operate directly over the string and measure the length and interaction of certain dependencies. There are also metrics that consider more than one isolated structure: surprisal and entropy reduction (Hale 2001, 2003, 2011; Levy 2013), for

[3] Following the advice of two reviewers, we refrain from using Graf *et al.*'s (2015) term *parsing metric*, which already has an established but distinct meaning in the formal parsing community.

instance, measure how the search space shrinks and grows during incremental processing.

The open-endedness of complexity metrics reflects the fact that the number of conceivable linking theories between the parser and the observed processing phenomena is dauntingly large. In the face of such an overabundance of choices, the methodologically soundest position is to explore simple metrics before moving on to more complicated ones. This is the stance we adopt throughout this paper. The MG parser has already been simplified to a degree where all ambiguity is abstracted away and parsing is reduced to index/outdex annotations of derivation trees. Sticking with our focus on derivation trees and maximal simplicity, we only consider complexity metrics that predict processing difficulty based on how the geometry of derivation trees affects memory usage.

That processing difficulty correlates with memory usage is a very common hypothesis in the psycholinguistic literature. The idea can be traced back to Kaplan (1974) and Wanner and Maratsos (1978),[4] with Joshi (1990), Gibson (1998, 2000) and many other as more recent examples (see Gerth (2015, Section 2.3.1) for a detailed discussion). Memory usage may be measured in many different ways, though, and as a result there is a myriad conceivable complexity metrics that differ only in minor details. This paper compares over a thousand memory-based complexity metrics, but fortunately they can be reduced to three basic concepts, which will be gradually refined and modified as we go along.

As we briefly remarked in Section 2.3, the MG parser does not actually hold nodes of the derivation tree in memory but rather *parse items* that encode various pieces of information about each node, in particular whether the node is the root of a subtree containing movers. For a parser that has to hold such parse items in memory, one can distinguish at least three kinds of memory usage:

Tenure How long is the item kept in memory?

Payload How many items are held in memory?

Size How many bits does the item consume in memory?

[4] We thank an anonymous reviewer for bringing these early works to our attention.

Each category is part of some complexity metric that has been invoked in previous work on MG parsing (Kobele *et al.* 2013; Graf and Marcinek 2014; Graf *et al.* 2015; Gerth 2015). In terms of annotated derivation trees, the three notions can be formalized as follows:

Definition 2 *Let t be some Minimalist derivation tree annotated with indices and outdices.*

- *For every node n with index i and outdex o ($i < o$), its tenure* $\text{ten}(n)$ *is $o - i$. A node's tenure is trivial iff* $\text{ten}(n) \leq 2$.
- *The payload of t is equal to the number of nodes in t with non-trivial tenure:* $|\{n \mid \text{ten}(n) > 2\}|$.
- *For every node n its size is identical to the number of phrases that are reflexively dominated by n, distinct from n, and are associated to a Move node that reflexively dominates n.*

For a concrete example, consider again the derivation tree in Figure 7. The tenure of the empty C-head is $11-3 = 8$, whereas the tenure of TP is just $4-3 = 1$. The derivation tree's payload is 5 as there are five nodes with non-trivial tenure (indicated by boxed outdices): the empty C-, T-, and v-heads, as well as DP and *likes*. The size of a parse item corresponding to node n is the same as the number of nodes below n that have a movement arrow pointing to somewhere above n. So the size of CP and v' is 1 and the size of T′ is 2, whereas the size of DP and v is 0.

The definition of size may strike the reader as very stipulative. It derives from how information about movers is stored by Stabler's (2013) top-down parser. For a detailed discussion, the reader is referred to Graf *et al.* (2015). Similarly, readers may wonder why the threshold for payload is set to 2 rather than 1. Once again this is done for technical reasons, discussed in Graf and Marcinek (2014).

3.2 *From memory usage to complexity metrics*

Note that tenure, size and payload are not exactly on equal footing. While payload is a property of derivation trees, tenure and size are properties of individual nodes/parse items. Consequently, payload can already be used as a complexity metric for our simple purposes: given two derivation trees, the one with lower payload is predicted to be easier to process. Graf and Marcinek (2014) use the name **Box** to distinguish payload as a complexity metric over derivation trees from

payload as general concept of memory usage. The name is motivated by the notational convention to draw a box around the outdices of nodes with non-trivial tenure, which we also adhere to in this paper. In contrast to payload, tenure and size can be applied to derivation trees in multiple ways.

Tenure was incorporated into three distinct complexity metrics by Kobele *et al.* (2013). Let T be the set of nodes of derivation tree t. Then

MaxT $\max(\{\text{ten}(n) \mid n \in T\})$
SumT $\sum_{n \in T, \text{ten}(n) > 2} \text{ten}(n)$
AvgT $\frac{\textbf{SumT}(t)}{\textbf{Box}(t)}$

So **MaxT** reports the maximum memory usage used by any single one node, **SumT** the total (non-trivial) tenure of the entire derivation tree, and **AvgT** the average memory usage of a node with non-trivial tenure. Recall that the derivation in Figure 1 has a payload of 5, which is also its **Box** value. Moreover we have **MaxT** $= 10$ (due to T), **SumT** $= 8 + 10 + 5 + 8 + 8 = 39$ (summing the tenure of all boxed nodes), and **AvgT** $= \frac{39}{5} = 7.8$.

Graf *et al.* (2015) furthermore generalize size to the complexity metric **Gap** in a fashion that mirrors **SumT** for tenure. To highlight this similarity, we call this metric **SumS** instead of **Gap** (the original name is motivated by the parallels to measuring the length of filler-gap dependencies). Let M be the set of all nodes of derivation tree t that are the root of a subtree undergoing movement. Also, for each $m \in M$, $i(m)$ is the index of m and $f(m)$ is the index of the highest Move node that m's subtree is moved to. With the visual aids in our derivation trees, M can be taken to consist of exactly those nodes that are the starting point of an arrow, while $f(m)$ is the target node of the highest arrow that starts in m. **SumS** sums the differences between these indices.

SumS $\sum_{m \in M} i(m) - f(m)$

Considering once more the derivation tree in Figure 1, we see that $M = \{\text{DP}, \text{John}\}$, $i(\text{DP}) - f(\text{DP}) = 7 - 3 = 4$, and $i(\text{John}) - f(\text{John}) = 9 - 1 = 8$. So the whole derivation tree has a **SumS** value of 12. The motivation behind **SumS** is again hard to convey without drilling deep into the bowels of the MG top-down parser. Intuitively, though, **SumS**

expresses the idea (independently argued for in O'Grady 2011) that moving a subtree is computationally expensive – the longer it takes to actually get to the subtree that needs to be moved, the higher the resource cost.

Even though **SumS** is transparently a size-based analog of **SumT**, no complexity metrics have been previously proposed for size that operate similar to **Box**, **MaxT** or **AvgT**. We introduce these metrics here for the sake of completeness, even though they will eventually turn out to be inadequate for sentence processing.

Movers $|M|$, where M is the set of all nodes that are the root of a subtree undergoing movement

MaxS $\max(\{i(n) - f(n) \mid n \in T\})$, where T is the set of all nodes of the derivation tree

AvgS $\frac{\textbf{SumS}(t)}{\textbf{Movers}(t)}$

For the example in Figure 7, we have **Movers** $= 2$ (only subject and object move), **MaxS** $= 8$ (topicalization of *John*), and **AvgS** $= \frac{(9-1)+(7-3)}{2} = 6$.

3.3 *Further refinements*

3.3.1 Recursive application

Another metric briefly mentioned in Graf and Marcinek (2014) is $\textbf{MaxT}^{\textbf{R}}$, which applies **MaxT** recursively. With **MaxT**, two derivation trees may receive exactly the same score and would thus be predicted to be equally difficult. $\textbf{MaxT}^{\textbf{R}}$ instead assigns each derivation tree a weight that enumerates in decreasing order the tenure of all nodes in the payload. Then derivation u is easier than derivation v iff their weights are identical up to position i, at which point u's weight contains a smaller number than v's weight. A similar strategy can also be used for size, yielding the complexity metric $\textbf{MaxS}^{\textbf{R}}$.

Our example derivation tree has the values $\textbf{MaxT}^{\textbf{R}} = [10, 8, 8, 8, 5]$ and $\textbf{MaxS}^{\textbf{R}} = [8, 4]$. Therefore it would be harder than a competing derivation with $\textbf{MaxT}^{\textbf{R}} = [10, 8, 8, 5]$, but easier than one with $\textbf{MaxS}^{\textbf{R}} = [8, 3]$.

3.3.2 Restriction by node type

Graf and Marcinek (2014) refine the set of metrics even more by relativizing them to specific types of nodes. For each metric **M**, an additional four variants $\mathbf{M_I}$, $\mathbf{M_L}$, $\mathbf{M_P}$, and $\mathbf{M_U}$ are defined.

$\mathbf{M_I}$ restriction of metric **M** to interior nodes

$\mathbf{M_L}$ restriction of metric **M** to leaf nodes

$\mathbf{M_P}$ restriction of metric **M** to pronounced leaf nodes

$\mathbf{M_U}$ restriction of metric **M** to unpronounced leaf nodes

For instance, the unrestricted **MaxT** value of our derivation was 10, but the refined values are $\mathbf{MaxT_I} = 5$ (on DP), $\mathbf{MaxT_L} = 10$ (on T), $\mathbf{MaxT_P} = 8$ (on *likes*), and $\mathbf{MaxT_U} = 10$ (on T).

Note that these restrictions make little sense for size-based metrics since moving subtrees usually contain pronounced material and the corresponding Move node is necessarily an interior node. Therefore we do not consider type-based restrictions of size metrics. At this point, then, the set of defined metrics includes **Box**, **MaxT**, **SumT**, **AvgT**, $\mathbf{MaxT^R}$, four restricted subtypes for each one of them, as well as the size-based metrics **Movers**, **MaxS**, **SumS**, **AvgS**, and $\mathbf{MaxS^R}$ (for a total of 30 metrics).

3.3.3 Time course of memory usage

The final metric to be considered refines payload so that it reflects maximum memory usage more faithfully. As we saw earlier, **Box** simply reports how many parse items had to be held in memory. However, a high **Box** value need not imply a heavy memory burden as long as one item is removed from memory before the next one is inserted. That is to say, if nodes u and v contribute to the payload of derivation t but the outdex of u is less than the index of v, then the two never reside in memory at the same time. In order to home in on this aspect, we define two metrics *convergence* **Con** and *divergence* **Div** that keep track of how many distinct nodes do or do not reside in memory at the same time.

Con $|\{\langle u, v\rangle \mid \text{ten}(u) \geq 2, \text{ten}(v) \geq 2, \text{index}(u) \leq \text{index}(v) \leq \text{outdex}(u)\}|$

Div $|\{\langle u, v\rangle \mid \text{ten}(u) \geq 2, \text{ten}(v) \geq 2, \text{outdex}(u) < \text{index}(v)\}|$

As before these metrics can be relativized to the four subtypes **I**, **L**, **P**, and **U**. Returning one final time to the derivation in Figure 7, we see that $\mathbf{Con_U} = |\{\langle \mathrm{C}, \mathrm{T}\rangle, \langle \mathrm{C}, v\rangle, \langle \mathrm{T}, v\rangle\}| = 3$ and $\mathbf{Div} = |\emptyset| = 0$.

3.3.4 Ranked complexity metrics

With just a handful of psycholinguistically plausible factors such as maximum and average memory usage and restrictions to specific types of nodes the number of metrics has quickly risen to a bewildering degree. But things do not stop here. Graf *et al.* (2015) argue in favor of a combined metric **MaxT** + **SumS** which uses **MaxT** to predict processing difficulty but relies on **SumS** whenever **MaxT** results in a tie. So given two derivation trees t_1 and t_2, t_1 is predicted to be easier than t_2 if either t_1 has a lower **MaxT** or t_1 and t_2 have the same **MaxT** value but t_1 has a lower **SumS** value. This is similar to constraint ranking in OT (Prince and Smolensky 2004), where a lower ranked constraint matters only if all higher ranked constraints have failed to pick out a unique winner. If complexity metrics are allowed to be ranked in such a way, their number quickly reaches an astronomical size. We have introduced 40 metrics, wherefore a ranked complexity metric can consist of up to 40 metrics. It follows that there are over 40 factorial (40!) distinct metrics that are ranked combinations of our 40 basic metrics – a truly astounding number. Even if one only allows pairs of our 40 complexity metrics, there are 1,600 distinct metrics (pairs of the form $\langle m, m\rangle$ are equivalent to just the metric m).

Ranked Metric Given a set C of complexity metrics, a ranked metric is an n-tuple $\langle c_1, \ldots, c_n\rangle$ such that for $1 \leq i, j \leq n$ it holds that $c_i \in C$ and that $i \neq j$ implies $c_i \neq c_j$. Given a ranked metric $\langle c_1, \ldots, c_n\rangle$ and two derivation trees t_1 and t_2, t_1 is predicted to be easier than t_2 iff there is some $j \leq n$ such that $c_i(t_1) = c_i(t_2)$ for all $i \leq j$ and c_j predicts t_1 to be easier than t_2.

3.4 *Discussion*

The large number of metrics poses a significant problem. Remember the promise of the MG parser and the psycholinguistic modeling work that builds on it: processing phenomena are explained in terms of the syntactic structures they involve, and in the other direction, syntactic analyses can be evaluated based on their processing predictions.

But the processing claims of these models arise from the interaction of three factors, which are the parser (represented via index/outdex annotation), the syntactic analysis (in the form of derivation trees), and the complexity metric.

There are few alternatives to the current top-down parser. Despite some suggestive evidence such as merely local syntactic coherence effects (Tabor *et al.* 2004; Konieczny 2005; Konieczny *et al.* 2009; Bicknell *et al.* 2009), there is still a large consensus among psycholinguists that if the human parser builds any kind of tree structures, it does not do so in a pure bottom-up fashion. The other prominent option is left-corner parsing. Unfortunately, no left-corner parser exists for MGs at this time because the notion *left corner* does not carry over neatly from CFGs (but see Hunter 2015b). Without a readily available alternative to top-down parsing, the two major parameters in the model are the choice of metric and the choice of syntactic analyses. But the larger the set of metrics, the higher the risk that just about any syntactic analysis will make the right predictions with some metric. This would significantly weaken the link between syntactic structure and processing effects, which is the very heart of the work carried out by Kobele *et al.* (2013), Graf and Marcinek (2014), Graf *et al.* (2015), and Gerth (2015).

Fortunately, this worst-case scenario does not seem to arise. It turns out that a few constructions involving relative clauses are sufficient to rule out the vast majority of these metrics. We have no principled explanation as to why this is the case – it is far from a logical necessity. But this result, established in the next section, strengthens the viability of the enterprise started by Kobele *et al.* (2013) to model processing phenomena with MGs and use these findings to distinguish competing Minimalist analyses. It demonstrates that I) a very simplified processing model can still account for a noteworthy range of challenging processing phenomena, and II) the set of workable complexity metrics is small enough to give the model discriminative power with respect to syntactic analyses.

4 TESTING METRICS WITH RELATIVE CLAUSES

Now that we have a precisely defined parsing model (abstractly represented in terms of annotated derivation trees) as well as a collection of

complexity metrics that link the parser's behavior to processing predictions, we are finally in a position to investigate how well these tools model a collection of phenomena from human sentence processing. All these phenomena, which will be presented in detail in Section 4.1, involve relative clauses (RCs) to some extent and have been studied separately in Kobele *et al.* (2013), Graf and Marcinek (2014), and Graf *et al.* (2015).[5] In contrast, we consider the full data set and test our much bigger collection of metrics against each one of them. We furthermore compare two competing analyses of RCs (4.2) using a fairly simple methodology of automated comparisons (4.3). We conclude that this small set of data points is highly discriminative in that it rules out a large number of metrics for each analysis (4.4) while still allowing for linguistically natural explanations of the observed patterns (4.5).

4.1 *Overview of relative clause constructions*

Our testing data for the comparison of metrics and syntactic proposals relies on several well-known processing contrasts involving RCs. RCs are a promising test case because they are complex enough to allow for syntactically interesting structures while factoring out aspects that aren't purely structural in nature such as co-reference resolution. The general idea is to take a pair of constructions A and B such that A is easier to process than B. This result then has to be replicated by the complexity metrics given a specific analysis of RCs.

The specific behavioral contrasts to be accounted for were chosen according to several criteria. First, the processing effects must be well-documented in the psycholinguistic research. Second, the phenomenon should involve a clear structural contrast, rather than just a meaning contrast (e.g. pronoun resolution). Third, ambiguity should not be a major factor, which rules out garden path effects.

1. **SC/RC < RC/SC**
   A sentence with a relative clause embedded inside a sentential complement (SC/RC) is easier to parse than a sentence with a

[5] Gerth (2015) investigates some additional phenomena which were not included in our data sample as we were not aware of her findings until recently, unfortunately.

sentential complement embedded inside a relative clause (RC/SC; Gibson 1998, 2000).

2. **SRC < ORC**
   - Subject relative clauses (SRCs) are easier to parse than object relative clauses (ORCs) in languages like English, where relative clauses are post-nominal and therefore follow their head noun (Mecklinger *et al.* 1995; Gibson 1998, 2000; Mak *et al.* 2002, 2006; Gordon *et al.* 2006).
   - SRCs are also easier to parse than ORCs in Chinese, Korean, and Japanese, where relative clauses are pre-nominal, that is to say, they precede their head noun (Miyamoto and Nakamura 2003, 2013; Lin and Bever 2006; Ueno and Garnsey 2008; Kwon *et al.* 2010; Gibson and Wu 2013).
3. **Right < Center**
   Right embedding is easier than center embedding.

These generalizations have been carefully established in the literature via self-paced reading experiments and ERP studies with minimal pairs such as the ones listed in (1)–(6).

(1) **SC/RC < RC/SC**

a. The fact [$_{\text{SC}}$ that the employee$_i$ [$_{\text{RC}}$ who the manager hired t_i] stole office supplies] worried the executive.

b. The executive$_i$ [$_{\text{RC}}$ who the fact [$_{\text{SC}}$ that the employee stole offices supplies] worried t_i] hired the manager.

(2) **SRC < ORC in English**

a. The reporter$_i$ [$_{\text{RC}}$ who t_i attacked the senator] admitted the error.

b. The reporter$_i$ [$_{\text{RC}}$ who the senator attacked t_i] admitted the error.

(3) **SRC < ORC in Chinese**

a. [$_{\text{RC}}$ t_i gongji yiyuan] de jizhe chengren-le cuowu
attack senator REL reporter admit-PRF error

b. [$_{\text{RC}}$ yiyuan gongji t_i] de jizhe chengren-le cuowu
senator attack REL reporter admit-PRF error

(4) **SRC < ORC in Korean**

a. [$_{RC}$ t_i uywon-ul kongkyekha-n] kica$_i$-ka
senator-ACC attacked-REL reporter-NOM
silswu-lul siinhayssta
error-ACC admitted

b. [$_{RC}$ uywon-i t_i kongkyekha-n] kica$_i$-ka
senator-NOM attacked-REL reporter-NOM
silswu-lul siinhayssta
error-ACC admitted

(5) **SRC < ORC in Japanese**

a. [$_{RC}$ t_i giin-ga hinanshita] kisha$_i$-ga ayamari-o
senator-ACC attacked reporter-NOM error-ACC
mitometa
admitted

b. [$_{RC}$ giin-ga t_i hinanshita] kisha$_i$-ga ayamari-o
senator-NOM attacked reporter-NOM error-ACC
mitometa
admitted

(6) **Right < Center**

a. The boy disappeared [$_{RC}$ who the man saw [$_{RC}$ who the woman praised]].

b. The boy [$_{RC}$ who the man [$_{RC}$ who the woman praised] saw] disappeared.

It should be pointed out that the SRC preference is less robust in Chinese than Korean or Japanese. This has been attributed to structural ambiguities (Gibson and Wu 2013), which is corroborated by Yun *et al.* (2014) and their ambiguity-based account rooted in entropy reduction. Recall from Section 2.3, though, that we deliberately ignore ambiguity in this paper so that only tree-geometric aspects of the derivation can derive processing effects. For this reason, we assume that Chinese would also exhibit a uniform preference for SRCs over ORCs if it were not for the confound of structural ambiguity.

Some of the contrasts above have previously proven difficult to account for. While the preference for SC/RC and SRC in English can be explained by string-based models such as the Dependency Locality Theory (Gibson 1998) or the Active-Filler strategy (Frazier 1987),

these models erroneously derive an ORC preference for East Asian languages with their pre-nominal RCs. This is because the head noun is closer to the object than the subject position of the RC in this case. A functional account like Keenan and Comrie's (1977) accessibility hierarchy, on the other hand, derives the SRC preference across languages but has little to say about the ease of SC/RC in comparison to RC/SC. That right embeddings are much easier than center embeddings has an elegant explanation in terms of left-corner parsing (Resnik 1992), but this account in turn does not generalize to the other configurations. Overall, then, the data points above have been accounted for individually, but their unification is challenging.

4.2 *Promotion and wh-analysis of relative clauses*

As one of the promises of the MG processing model is the ability to distinguish syntactic analyses based on their processing predictions, our evaluation uses two popular proposals for the structure of RCs: the *wh-movement* analysis (Chomsky 1965, 1977; Montague 1970; Heim and Kratzer 1998), and the *promotion* analysis (Vergnaud 1974; Kayne 1994). Graf *et al.* (2015) did the same in their investigation of the SRC preference in East Asian, whereas Kobele *et al.* (2013) and Graf and Marcinek (2014) only used a promotion analysis.

Both the promotion analysis and the wh-analysis posit that the gap inside the RC is initially filled by some element, but disagree on what this element is and where it moves. In the promotion analysis, it is the head noun itself that starts from the gap position. The wh-analysis has two variants. Either the relative pronoun[6] moves from the gap position, or it acts as the C-head of the RC while a silent operator undergoes movement from the base position. For the purposes of this paper the two variants of the wh-movement analysis are fully equivalent.

[6] The use of "relative pronoun" is slightly misleading here in that the relative clause markers in Chinese and Korean are not pronouns (as is rightfully noted by an anonymous reviewer). But since the syntactic category of LIs is ignored by all complexity metrics, we freely change between the terms relative pronoun and RC marker in the discussion. We also represent the East Asian RC markers with *who* in the derivation trees in an attempt to ease the comparison to the English derivation trees.

Notably absent are proposals that do not involve any movement at all. This is because in the absence of movement, the MG parser behaves exactly like a recursive descent parser for CFGs and thus would have little new to offer. In addition, the comparison and detailed analysis of the complexity metrics already involves a multitude of factors, so that increasing the number of analyses would run the risk of rendering the discussion (even more) impenetrable.

With both the promotion analysis and the wh-analysis, the target of movement depends on whether RCs are post-nominal or pre-nominal in the language under investigation. Let us consider languages with post-nominal RCs like English, French, and German. All these languages also have overt complementizers, although they may optionally remain unpronounced, as is the case in English. The general template is [$_{DP}$ Det head-noun [$_{RC}$ complementizer subject verb object]], with either the subject or the object unrealized. The position of the verb depends on language-specific word order constraints, but we can safely ignore this aspect because English is the only language with post-nominal RCs in our data set. Figures 8 and 9 show the promotion analysis and the wh-analysis, respectively, for the SRC *The reporter who attacked the senator admitted the error*. In both derivation trees the element that fills the gap in the SRC moves to the CP specifier (Spec,CP), i.e. the left edge of the relative clause. But note that the head noun is outside the RC in the wh-movement analysis, whereas it is in Spec,CP (and thus inside the RC) in the promotion analysis.

The only difference between SRC and ORC under these analyses is the position that the mover occupies initially. In the SRC, the mover fills the base position of subjects (equated with Spec,*v*P here), whereas the ORC requires the mover to start out in object position (i.e. as the VP complement). This is illustrated in Figure 10, which depicts an ORC with an embedded sentential complement.

Languages with pre-nominal RCs, such as Chinese, Japanese, and Korean, can also be accommodated, but the word order differences render both analyses more complex. Below is an example of pseudo-English SRCs and ORCs with Chinese word order.

(7) a. [$_{DP}$ [$_{RC}$ _ invited the tycoon who] the mayor] likes wine.

b. [$_{DP}$ [$_{RC}$ the tycoon invited _ who] the mayor] likes wine.

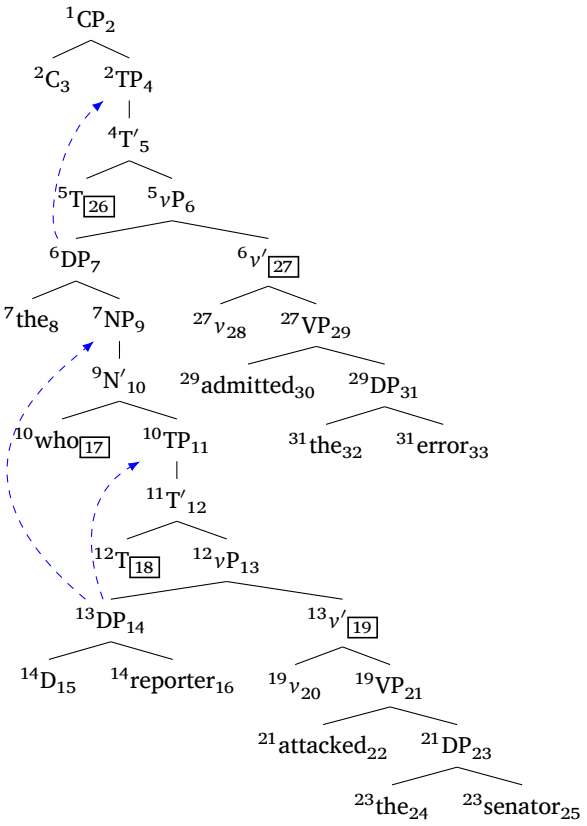


Figure 8: English SRC, promotion analysis

Since standard MGs do not provide a headedness parameter to determine the linearization of arguments (following the received view in Minimalist syntax), the pre-nominal word order must be derived from the post-nominal one via movement. This causes the wh-movement analysis and the promotion analysis to diverge more noticeably.

In the promotion analysis, the RC is no longer a CP, but rather a RelP that contains a CP (see also Yun *et al.* 2014). The head noun still moves from within the RC to Spec,CP, but this is followed by the TP moving to Spec,RelP. This creates the desired word order with the complementizer between the rest of the RC in Spec,RelP and the head noun in Spec,CP. In the wh-movement analysis, on the other hand, the head noun is once again outside the RC, which is just a CP instead of a RelP. The complementizer starts out in subject or object position

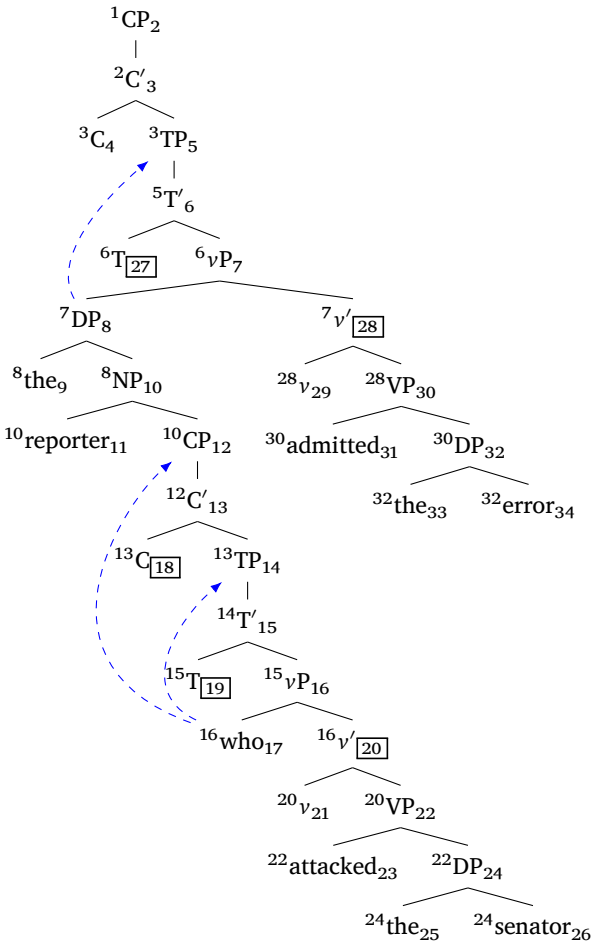


Figure 9: English SRC, wh-movement analysis

depending on the type of RC, and then moves into a right specifier of the CP (rightward movement is not part of Stabler's (2013) MG parser, but we can easily add it without modifying the annotation rules from Definition 1 as they are defined in terms of s-precedence). The CP subsequently moves to the specifier of the DP of the head noun, once again yielding the desired word order with the complementizer between the RC and the head noun. In sum, the promotion analysis needs to posit a new phrase RelP but all movement is leftward and takes place within this phrase. This contrasts with the wh-movement

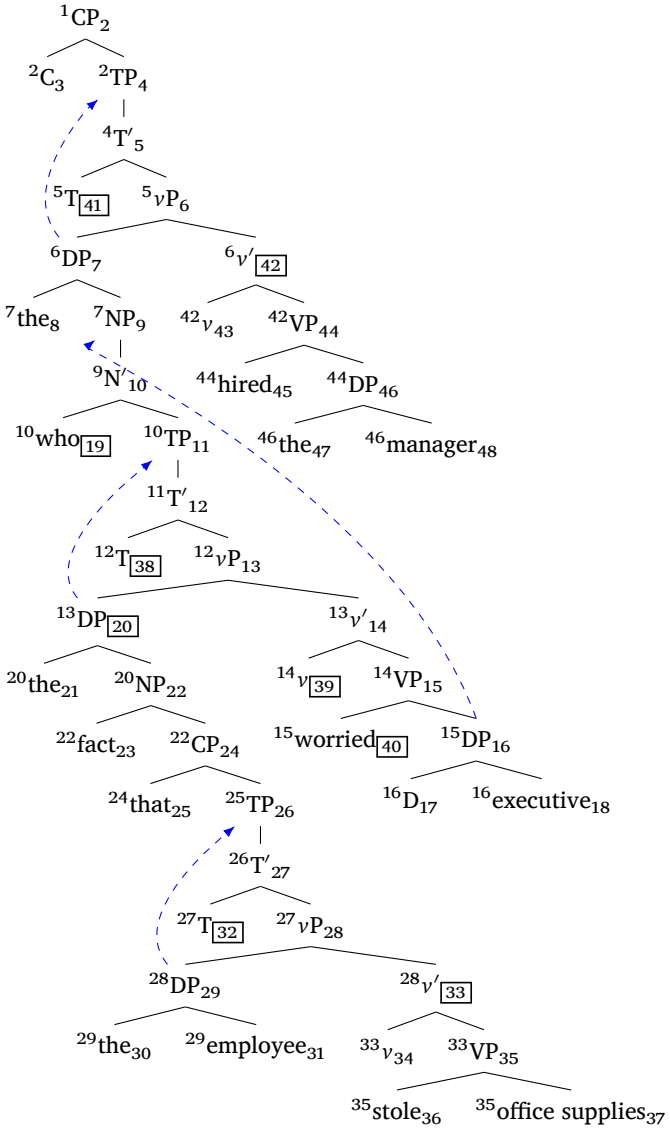


Figure 10: ORC containing a sentential complement, promotion analysis

analysis, which sticks with a single CP but invokes one instance of rightward movement and moves the RC into Spec,DP, a higher position than Spec,RelP. Examples of the two derivation trees for a Chinese SRC are given in Figures 11 and 12, where dotted arrows are used instead of dashed ones for rightward movement.

Among the three East Asian languages, Chinese still has the simpler analysis thanks to its SVO word order, whereas Japanese and Korean are SOV languages. As was the case with the linear order of RCs relative to their head noun, Minimalist syntax assumes that the SOV word order is derived via Move rather than simply linearizing the complement of the verb to its left. The standard assumption is that SOV

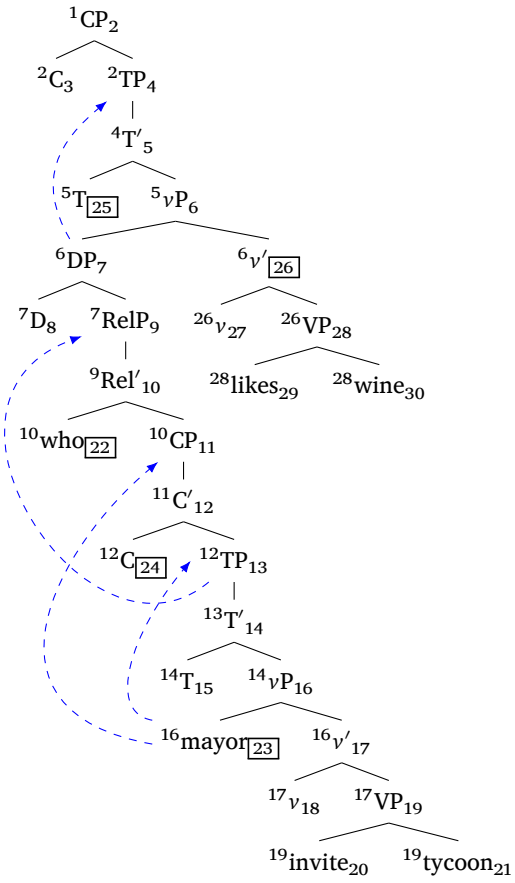


Figure 11: SRC in Chinese, promotion analysis

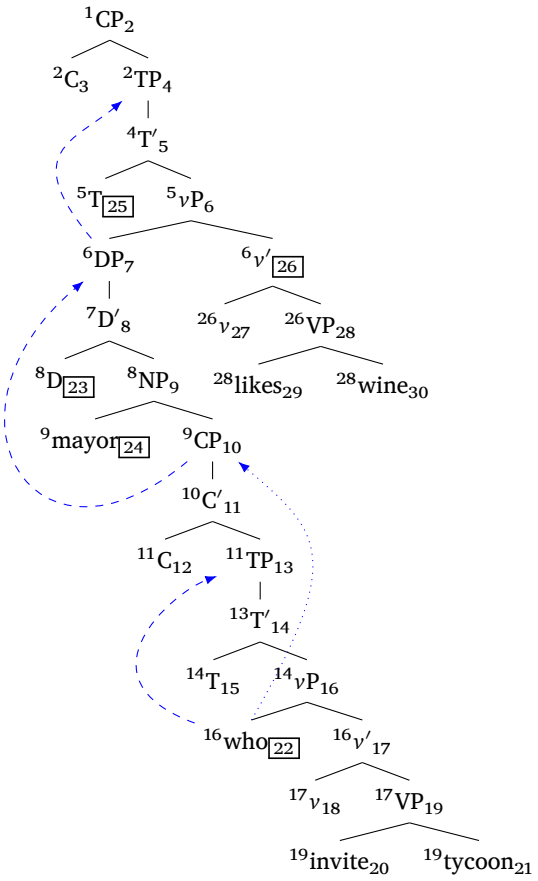


Figure 12: SRC in Chinese, wh-movement analysis

languages require the object to move from the VP-complement position to a specifier of *v*P as exemplified in Figure 13. While this might seem like a minor complication, we will see in the next section that it actually causes many metrics to incorrectly prefer ORCs over SRCs. Korean and Japanese thus show that the complexity metrics are indeed exquisitely sensitive to minor structural alterations.

We also use rightward movement in right embedding constructions (Figure 14), as embedding without additional movement yields center embedding structures. Whether these instances of extraposition are best analyzed as rightward movement has been called into question in recent research (Hunter and Frank 2014), but it is the best

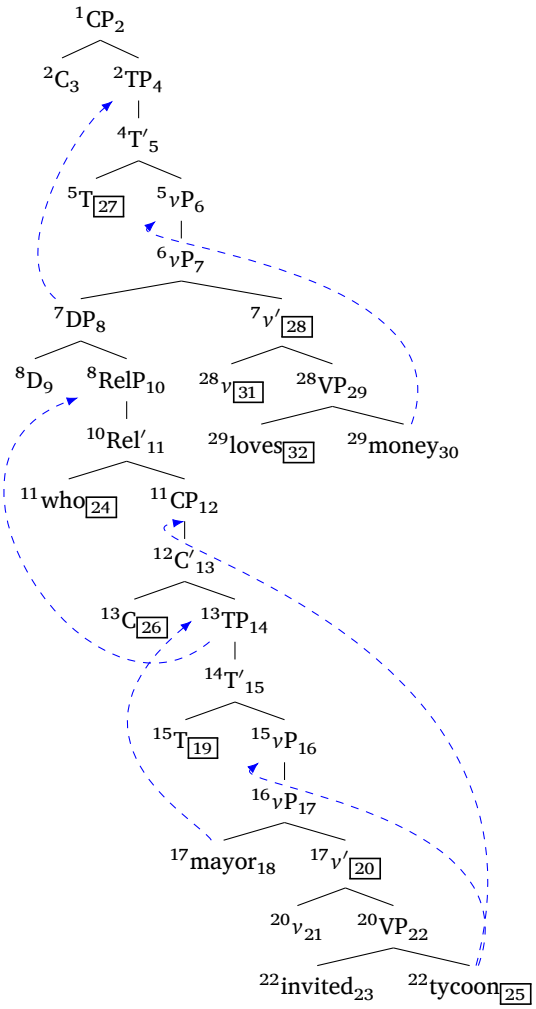


Figure 13: ORC in Korean, promotion analysis

choice here to maintain analytical consistency across the different constructions.

For a full listing of all the analyses with annotated derivation trees, the reader is referred to the supplementary material for this article. Derivation trees for Japanese are omitted since they are identical to the Korean ones except that the RC complementizer remains unpronounced.

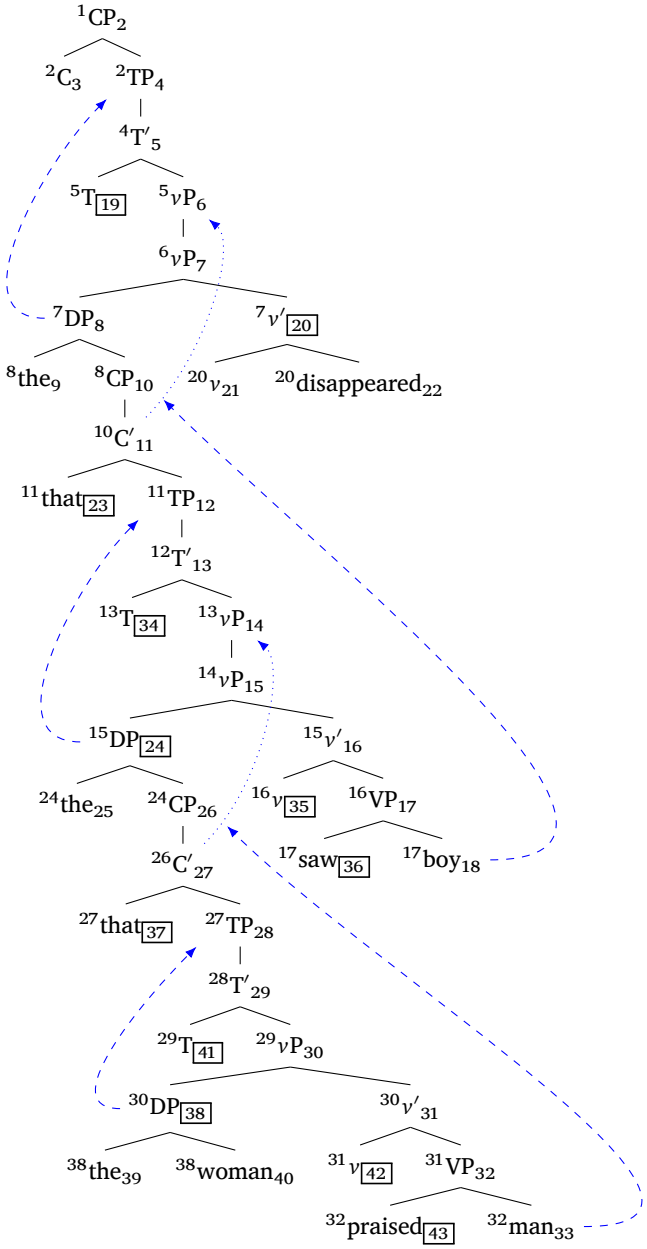


Figure 14: Right embedding, promotion analysis

4.3 *Methodological remarks*

As 1,600 metrics cannot be accurately compared by hand, we rely on a collection of Python scripts, available in the Github repository of the Stony Brook Computational Linguistics lab: `https://github.com/CompLab-StonyBrook`. For each basic metric, these scripts perform pairwise comparisons of minimally different derivation trees, e.g. the English SRC in Figure 8 and its ORC counterpart. Whichever one receives a better (= lower) score has a lower memory burden is thus predicted to be easier to process. From the relative rankings that are obtained this way one can then automatically compute all the metrics – including combinations of multiple metrics – that correctly predict all processing contrasts.

Note that the difficulty metric only has to account for overall sentence difficulty. This is different from more ambitious approaches such as Hale (2001) and Yun *et al.* (2014), which seek to predict online difficulty, i.e. how difficulty increases or decreases with each word in the input. Modeling online processing is feasible with certain complexity metrics like **MaxT** (see Kobele *et al.* 2013 and Gerth 2015), but it is hard to automatically compare metrics at this level of granularity. Finally, we reiterate that all ambiguity is factored out – we only consider how the parser builds a specific derivation tree, rather than how it finds this tree among many alternatives.

4.4 *Quantitative evaluation of complexity metrics*

The performance of the basic metrics with the respective syntactic analyses is summarized in Tables 1 and 2. A checkmark (✓) indicates that the metric correctly predicts structure A to be easier than structure B, a tie that they are expected to be equally difficult, and a cross (✗) that the complexity metric incorrectly reverses difficulty, making B easier than A. Consequently, all basic metrics that contain a cross in at least one column can be discarded. This leaves only one metric for the promotion analysis – $\mathbf{MaxT}^{\mathbf{R}}_{\mathbf{U}}$ – and one for the wh-movement analysis – $\mathbf{AvgT}_{\mathbf{P}}$.

Many inadequate basic metrics, though, may still occur as the second component in a ranked metric. As the second component is only invoked to handle ties for the first component, wrong predictions for a given contrast have no effect unless the first component could not conclusively resolve this contrast. When ranked metrics are also taken

Table 1: Predictions of complexity metrics with promotion analysis

	SC/RC < RC/SC	SRC < ORC Eng	Chi	Kor	Jap	Right < Center
Box	tie	✓	✓	tie	tie	✗
Box$_{I}$	tie	tie	✓	✓	✓	✗
Box$_{L}$	tie	✓	✓	✗	✗	tie
Box$_{P}$	tie	✓	tie	✗	✗	✓
Box$_{U}$	✓	✓	✓	tie	tie	✗
AvgT	✓	✓	✗	✗	✗	✓
AvgT$_{I}$	✗	✓	✗	✗	✗	✗
AvgT$_{L}$	✓	✓	✗	✓	✓	✓
AvgT$_{P}$	✓	✓	✗	✗	✗	✓
AvgT$_{U}$	✓	✓	✗	✓	✓	✓
MaxT	tie	tie	tie	tie	tie	✓
MaxT$_{I}$	tie	tie	tie	tie	tie	✗
MaxT$_{L}$	tie	tie	tie	tie	tie	✓
MaxT$_{P}$	✓	✓	tie	tie	✗	✓
MaxT$_{U}$	tie	tie	tie	tie	tie	✓
MaxT^{R}	✓	✓	✗	✗	✗	✓
MaxT$^{R}_{I}$	✗	✓	✓	✓	✓	✗
MaxT$^{R}_{L}$	✓	✓	✗	✗	✗	✓
MaxT$^{R}_{P}$	✓	✓	✗	✗	✗	✓
MaxT$^{R}_{U}$	✓	✓	✓	✓	✓	✓
SumT	✓	✓	✓	✗	✗	✗
SumT$_{I}$	✗	✓	✓	✓	✓	✗
SumT$_{L}$	✓	✓	✗	✗	✗	✓
SumT$_{P}$	✓	✓	✗	✗	✗	✓
SumT$_{U}$	✓	✓	✓	✓	✓	✗
AvgS	✗	✓	✓	✓	✓	✗
Movers	tie	✓	✓	tie	tie	✗
MaxS	tie	✓	✓	tie	tie	✗
MaxS^{R}	✗	✓	✓	✓	✓	✗
SumS	✗	✓	✓	✓	✓	✗
Con	✗	✓	✓	✗	✗	✗
Con$_{I}$	✗	tie	✓	✓	✓	✗
Con$_{L}$	✗	✓	✓	✗	✗	✓
Con$_{P}$	tie	✓	tie	✗	✗	✓
Con$_{U}$	tie	✓	✓	tie	tie	✗
Div	✓	tie	✓	✓	✓	✗
Div$_{I}$	✓	tie	tie	tie	tie	✗
Div$_{L}$	✓	tie	✓	✗	✗	✗
Div$_{P}$	tie	tie	tie	✗	✗	tie
Div$_{U}$	tie	tie	tie	tie	tie	✗

	SC/RC < RC/SC	SRC < ORC Eng	SRC < ORC Chi	SRC < ORC Kor	SRC < ORC Jap	Right < Center
Box	tie	✓	✓	✗	✗	✗
Box$_{I}$	tie	tie	✓	✓	✓	✗
Box$_{L}$	tie	✓	tie	✗	✗	✗
Box$_{P}$	tie	✓	✗	✗	✗	tie
Box$_{U}$	tie	✓	✓	tie	✗	✗
AvgT	✓	✓	✗	✓	✓	✓
AvgT$_{I}$	✗	✓	✗	✗	✗	✗
AvgT$_{L}$	✓	✓	✗	✓	✓	✓
AvgT$_{P}$	✓	✓	✓	✓	✓	✓
AvgT$_{U}$	✓	✓	✗	✓	✓	✓
MaxT	tie	tie	tie	tie	tie	✓
MaxT$_{I}$	tie	tie	tie	tie	tie	✗
MaxT$_{L}$	tie	tie	tie	tie	tie	✓
MaxT$_{P}$	✓	✓	tie	tie	tie	✓
MaxT$_{U}$	tie	tie	tie	tie	tie	✓
MaxT^{R}	✓	✓	✗	✗	✗	✓
MaxT$^{R}_{I}$	✗	✓	✓	✓	✓	✗
MaxT$^{R}_{L}$	✓	✓	✗	✗	✗	✓
MaxT$^{R}_{P}$	✓	✓	✗	✗	✗	✓
MaxT$^{R}_{U}$	✓	✓	✓	✓	✗	✓
SumT	✓	✓	tie	✗	✗	✓
SumT$_{I}$	✗	✓	✓	✓	✓	✗
SumT$_{L}$	✓	✓	✗	✗	✗	✓
SumT$_{P}$	✓	✓	✗	✗	✗	✓
SumT$_{U}$	✓	✓	✓	✓	✗	✓
AvgS	✗	tie	✓	✓	✓	✗
Movers	tie	✓	✓	tie	tie	✗
MaxS	tie	✓	✓	tie	tie	✗
MaxS^{R}	✗	✓	✓	✓	✓	✗
SumS	✗	✓	✓	✓	✓	✗
Con	✗	✓	✗	✗	✗	✓
Con$_{I}$	✗	tie	✗	✓	✓	tie
Con$_{L}$	✗	✓	tie	✗	✗	✓
Con$_{P}$	tie	tie	✓	✗	✗	✓
Con$_{U}$	✗	✓	✗	tie	✗	✓
Div	✓	tie	tie	✗	✗	✗
Div$_{I}$	✓	tie	tie	tie	tie	✗
Div$_{L}$	✓	tie	tie	✗	✗	✗
Div$_{P}$	tie	tie	tie	✗	✗	✗
Div$_{U}$	✓	tie	tie	tie	✗	✗

Table 2: Predictions of complexity metrics with wh-movement analysis

Table 3: List of empirically viable complexity metrics

	Promotion	*Wh-Movement*
Basic	$\mathbf{MaxT}_{\mathbf{U}}^{\mathbf{R}}$	$\mathbf{AvgT}_{\mathbf{P}}$
Ranked	$\langle \mathbf{MaxT}, \mathbf{SumT}_{\mathbf{U}} \rangle$	$\langle \mathbf{MaxT}_{\mathbf{P}}, \mathbf{AvgS} \rangle$
	$\langle \mathbf{MaxT}_{\mathbf{L}}, \mathbf{SumT}_{\mathbf{U}} \rangle$	$\langle \mathbf{MaxT}_{\mathbf{P}}, \mathbf{Box}_{\mathbf{I}} \rangle$
	$\langle \mathbf{MaxT}_{\mathbf{U}}, \mathbf{SumT}_{\mathbf{U}} \rangle$	$\langle \mathbf{MaxT}_{\mathbf{P}}, \mathbf{Con}_{\mathbf{I}} \rangle$
		$\langle \mathbf{MaxT}_{\mathbf{P}}, \mathbf{MaxS}^{\mathbf{R}} \rangle$
		$\langle \mathbf{MaxT}_{\mathbf{P}}, \mathbf{MaxT}_{\mathbf{I}}^{\mathbf{R}} \rangle$
		$\langle \mathbf{MaxT}_{\mathbf{P}}, \mathbf{SumS} \rangle$
		$\langle \mathbf{MaxT}_{\mathbf{P}}, \mathbf{SumT}_{\mathbf{I}} \rangle$

into consideration, the number of metrics increases from 40 to 1,600. The number of empirically adequate metrics, on the other hand, does not increase by the same factor and grows from 1 to 4 (promotion) and 8 (wh-movement), respectively. No metric is a viable candidate for both analyses (see Table 3). Note that these numbers do not include ranked metrics whose first component is an empirically adequate basic metric ($\mathbf{MaxT}_{\mathbf{U}}^{\mathbf{R}}$ or $\mathbf{AvgT}_{\mathbf{P}}$) because the second metric would never be used in those cases. If those pair metrics are included, the respective numbers grow to $4 + 39 = 43$ and $8 + 39 = 47$. Depending on how one counts, then, between $\frac{4}{1600-39} = 0.2\%$ and $\frac{47}{1600} = 2.9\%$ of the full space of complexity metrics can account for the five observed processing contrasts with relative clauses. In addition, all the remaining ranked metrics have some variant of **MaxT** as their first component. This shows that the underspecification problem is not nearly as bad as one might expect, with a few contrasts ruling out the great majority of metrics.

In fact, the five constructions differ in their discriminatory power in a manner that roughly reflects how difficult they are to account for. For example, the preference for SRCs over ORCs in English requires no structure at all and can be explained purely in terms of string distance (Gibson 1998, 2000), and no metric reverses difficulty for this construction. Even the number of ties is comparatively low. The same contrast is much harder to account for in East Asian languages with their pre-nominal RCs. String-based explanations fail in this case, and so do more than half of all the basic metrics. The processing differ-

ence between right embedding and center embedding is interesting in this case because there are a variety of explanations in the psycholinguistic literature, and except for the size- or diversion-based metrics, all the core metrics have some variant that captures the contrast. The failure of size-based metrics is not surprising in this case. Recall that right embedded RCs induce additional syntactic complexity because they start out as center embedded RCs that have to undergo rightward movement. The additional movement steps inevitable cause size-based metrics to make the wrong predictions. Crucially, though, not all metrics fall into this trap, which proves that well-chosen complexity metrics can factor out irrelevant aspects of structural complexity.

4.5 *Qualitative evaluation of complexity metrics*

Since the connection between complexity metrics and the structure of derivation trees is very subtle and sensitive to even minor differences, determining why a complexity metric fails to capture a specific contrast while succeeding at another can be difficult. An exhaustive discussion of all the patterns reported in Table 1–3 is not feasible within the confines of a single paper. Instead, we present a few general observations on the role of **MaxT**, which has been a prominent metric in all previous work on MG parsing and is a component of almost all successful metrics.

First it is instructive, though, to consider why $\mathbf{AvgT_P}$ works for the wh-analysis but fails for the promotion analysis. The problematic constructions are the East-Asian RCs. Recall that in the promotion analysis, it is the head noun that moves from the gap, whereas in the wh-movement analysis it is the RC marker (simply transcribed as *who* in our derivation trees). Since RCs in East-Asian languages are prenominal and have the RC marker at their very end, the wh-movement analysis has

1. high tenure on the head noun outside the RC (which is encountered before the RC but cannot be finished until the latter is complete),
2. medium tenure on the RC marker in SRCs (as it occupies the structurally prominent subject position, which means that it is hypothesized early by the parser but must wait until the rest of the RC is completed),

3. low tenure on the verb in Korean and Japanese (which is introduced at the same time as the object but only finished after it due to object movement to the left).

In an ORC, the object moves to the right of the RC, so the low tenure on the RC marker disappears, and since it is an ORC the verb does not have any tenure either. But $\mathbf{AvgT_P}$ divides the sum of tenure of pronounced nodes by the number of pronounced nodes with non-trivial tenure. Removing two entries with low tenure ends up increasing the $\mathbf{AvgT_P}$ value for ORCs. The final numbers are $\frac{16+6+3}{3} = 8.3$ for SRCs in contrast to $\frac{16}{1} = 16$ for an ORC.

The structural differences in the promotion analysis, on the other hand, mean that although the switch from SRC to ORC reduces the tenure on the mover (the head noun, rather than the relative pronoun) it does not completely eliminate it. Hence the mover still counts towards the payload and thus greatly lowers the $\mathbf{AvgT_P}$ value for ORCs: $\frac{13+7+3}{2} = 11.5$ in contrast to $\frac{13+3}{2} = 8$. The success of $\mathbf{AvgT_P}$ with the wh-movement thus relies on completely eliminating non-trivial tenure on some nodes in ORCs, rather than just reducing it. The counterintuitive prediction of **AvgT** and its variants – if a derivation contains a node with high tenure, it will become easier the more nodes have low tenure instead of no tenure – accidentally makes the right prediction for SRCs and ORCs.

Let us now turn to **MaxT**, which strikes us as a more insightful and overall more robust choice of metric. The non-recursive variants of **MaxT** are a good choice for ranked metrics because they rarely make a completely wrong prediction but instead produce many ties. This is the reason why all successful ranked metrics contain them as their first component: a complexity metric with a cross in at least one column cannot be the first component of a ranked metric, which rules out all basic metrics except the "tie-heavy" **MaxT** variants (and the basic metrics that capture all the data, for which we do not list any ranked metrics).

The high frequency of ties with **MaxT** variants is a natural consequence of our focus on embedding constructions. All embedding constructions follow a template where different subtrees are inserted into a fixed main clause. For instance, the English SRC and ORC sentences differ only in the shape of their RCs; the main clause always has the

same structure. The overall number of steps it takes to parse an RC is independent of whether it is an SRC or an ORC. But this, in turn, implies that I) the nodes in the main clause that are introduced before the RC but cannot be worked on until the RC is finished (e.g. T and v' in Figure 8) have very large tenures exceeding that of any node inside the RC, and II) the tenure of these nodes is independent of whether the RC is an SRC or an ORC. As **MaxT** metrics only pay attention to the largest tenure in a tree, the differences between SRCs and ORCs get drowned out by the high tenure on nodes in the main clause.

This accidental flattening of contrasts does not occur in the case of right and center embedded RCs because the movement of an RC in right embedding directly interacts with the nodes in the clause containing the RC. In particular, moving an RC to the right of an LI l means that l can be worked on before the RC is explored by the parser, thus reducing its tenure. With center embedding, the parser would first have to explore the full RC, so the sister node of the RC would wind up with very high tenure. The overall generalization, then, is that **MaxT** metrics flatten contrasts where the differences between constructions are restricted to nodes within the embedded subtree.

$\mathbf{MaxT_P}$ is an exception because its restriction to pronounced nodes filters out the tenure of interior nodes like v' and unpronounced lexical heads like T. This improves its performance for the SC/RC versus RC/SC contrast as well as English SRC and ORC constructions. If our analysis had treated T as a pronounced head (e.g. for *do* support, or as the carrier of inflection that affix hops onto the verb), $\mathbf{MaxT_P}$ would also produce ties in these cases. But even in this case the behavior of $\mathbf{MaxT_P}$ could still be replicated by a metric that ignores interior nodes and functional heads, irrespective of whether they are pronounced.

While $\mathbf{MaxT_P}$ improves on other variants in some respects, it is also the only non-recursive **MaxT** version to incorrectly derive an ORC advantage in Japanese with the promotion analysis. This is due to the RC marker being unpronounced in Japanese, so that the only pronounced nodes with tenure are the head noun and the embedded verb. The head noun has the same tenure for SRC and ORC, but the embedded verb has non-trivial tenure in the SRC as it is introduced at the same time as the object but must wait for it to move leftward to Spec,vP. In the ORC, on the other hand, the object moves to a position to the right of the embedded verb, so that the latter can be completed

as soon as it is introduced. The ORC advantage thus is due to object extraction negating the inherent disadvantage of object movement.

In sum, it seems that any variant of **MaxT** that does not restrict itself to just pronounced nodes provides a solid baseline for a ranked metric with a suitably chosen ancillary metric to resolve ties. **MaxT** has previously been studied by Kobele *et al.* (2013), Graf and Marcinek (2014) and Gerth (2015), and it can even be traced back to Kaplan (1974) and Wanner and Maratsos (1978). It also plays a role in the TAG processing models of Joshi (1990) – in fact, Joshi (1990) ignores the memory usage of empty nodes and thus uses what amounts to our $\mathbf{MaxT_P}$, which is part of the majority of viable metrics. That three very different processing models home in on the same kind of memory usage as a benchmark for processing difficulty is very suggestive.

From the perspective of Minimalist syntax, $\langle \mathbf{MaxT}, \mathbf{SumS} \rangle$ and $\langle \mathbf{MaxT}, \mathbf{MaxS^R} \rangle$ are arguably the most elegant metric as they, intuitively speaking, combine maximum memory load with the total resource demand of all movement dependencies. In the generative literature, O'Grady (2011) has independently argued for the impact of movement dependencies on sentence processing, supporting a size-based metric. Our study confirms that these conceptually pleasing metrics have a lot of explanatory power to offer, although there are still some viable alternatives.

5 FURTHER OBSERVATIONS AND DISCUSSION

While the present study considers a much wider range of constructions and metrics than previous work on MG processing, it is still more limited in its scope than is desirable. The set of syntactic analyses, processing phenomena, and MG parsing algorithms all need to be extended to get a fuller picture of the empirical feasibility of this approach.

Our syntactic analyses still fix a plethora of parameters that need to be carefully modulated. For example, the low starting position of subjects and the movement of objects to Spec,*v*P cause tenure on T and *v*, respectively, which affects certain metrics. Replacing rightward movement by sequences of leftward movement (also known as remnant movement) will also be picked up on by some metrics, as would the introduction of a general headedness parameter to do away with certain movement steps. Since the derivation trees using these alter-

native proposals need to be carefully constructed by hand, a piecewise comparison that alters only one parameter at a time is a very laborious process.

A reviewer raises similar concerns and asks how useful these results are considering that the structure of East Asian languages is not nearly as well understood as that of English, wherefore their Minimalist analyses are much more likely to be fatally flawed. We agree that if push comes to shove, a metric's failure to account for the East Asian processing patterns has less weight than its performance on English data. However, the English data that is available and easily tested in this framework lacks some discriminatory power that the East Asian RC data provides. In science, we have to work with the data that is available, even if that data is sometimes sub-optimal.

But suppose that the structure of East Asian RCs does indeed need to be reanalyzed. We do not believe that this would lead to completely different metrics being chosen. We did some tests with an analysis of Korean and Japanese that simply linearizes the object to the left of the verb rather than moving it to Spec,*v*P. This made the processing predictions for them more similar to Chinese, and as a result widened the set of feasible metrics to also include ranked metrics whose first component is **SumT** for the wh-movement analysis or a variant of **Box** for the promotion analysis. Crucially, though, all the previously successful metrics were still available.

In the other direction, we also experimented with adding the preference for crossing dependencies over nested dependencies (Bach *et al.* 1986) to our data set. This preference was already shown in Kobele *et al.* (2013) to be predicted by **MaxT**. So it comes as little surprise that this contrast has no discriminative power relative to our current data set. All of our successful metrics correctly predict the contrast. Preliminary work on attachment preferences for dative arguments in Korean and quantifier scope preferences in English suggest that these, too, can be accounted for with the metrics identified in this paper. Overall, then, it seems that the class of complexity metrics carved out in this paper is fairly robust and more than just an accident.

CONCLUSION

We defined a large set of reasonably simple complexity metrics that make predictions about processing behavior based on the shape of index/outdex annotated MG derivation trees that closely mimic well-known analyses from Minimalist syntax. Only a few metrics could cover the full range of relative clause constructions, suggesting that the choice of metric is much more restricted than one might initially expect, and that underspecification is not too much of an issue in practice. In addition, the fact that it was at all possible to give a unified explanation of relative clause processing effects, which have proven challenging to deal with in the psycholinguistic literature, is encouraging. The MG processing model we advocate deliberately abstracts away from many aspects of sentence processing in order to clearly bring out the role that might be played by syntactic factors. It seems that at least in the case of relative clauses, structural considerations go a long way.

ACKNOWLEDGMENTS

We are grateful to the three anonymous reviewers, whose feedback has led to several changes in the presentation of the material (for the better, we like to believe). This research project would not exist if it were not for the continued encouragement by John Drury and the unique atmosphere created by the participants of the Fall 2014 parsing seminar at Stony Brook University.

REFERENCES

Emmon BACH, Colin BROWN, and William MARSLEN-WILSON (1986), Crossed and Nested Dependencies in German and Dutch: A Psycholinguistic Study, *Language and Cognitive Processes*, 1:249–262.

Klinton BICKNELL, Roger LEVY, and Vera DEMBERG (2009), Correcting the Incorrect: Local Coherence Effects Modeled with Prior Belief Update, in *Proceedings of the 35th Annual Meeting of the Berkeley Linguistics Society*, pp. 13–24, doi:10.3765/bls.v35i1.3594.

Noam CHOMSKY (1965), *Aspects of the Theory of Syntax*, MIT Press, Cambridge, MA.

Noam CHOMSKY (1977), *Essays on Form and Interpretation*, New York, North Holland.

Noam CHOMSKY (1995a), Bare Phrase Structure, in Gert WEBELHUTH, editor, *Government and Binding Theory and the Minimalist Program*, pp. 383–440, Blackwell, Oxford.

Noam CHOMSKY (1995b), *The Minimalist Program*, MIT Press, Cambridge, MA, doi:10.7551/mitpress/9780262527347.003.0003.

Noam CHOMSKY (2001), Derivation by Phase, in Michael J. KENSTOWICZ, editor, *Ken Hale: A Life in Language*, pp. 1–52, MIT Press, Cambridge, MA.

Philippe DE GROOTE (2001), Towards Abstract Categorial Grammars, in *Association for Computational Linguistics, 39th Annual Meeting and 10th Conference of the European Chapter*, pp. 148–155.

Meaghan FOWLIE (2013), Order and Optionality: Minimalist Grammars with Adjunction, in András KORNAI and Marco KUHLMANN, editors, *Proceedings of the 13th Meeting on the Mathematics of Language (MoL 13)*, pp. 12–20.

Lyn FRAZIER (1987), Sentence Processing: A Tutorial Review, in M. COLTHEART, editor, *Attention and Performance XII: The Psychology of Reading*, pp. 559–586, Lawrence Erlbaum Associates, Inc.

Werner FREY and Hans-Martin GÄRTNER (2002), On the Treatment of Scrambling and Adjunction in Minimalist grammars, in Gerhard JÄGER, Paola MONACHESI, Gerald PENN, and Shuly WINTNER, editors, *Proceedings of the Conference on Formal Grammar*, pp. 41–52.

Hans-Martin GÄRTNER and Jens MICHAELIS (2007), Some Remarks on Locality Conditions and Minimalist Grammars, in Uli SAUERLAND and Hans-Martin GÄRTNER, editors, *Interfaces + Recursion = Language? Chomsky's Minimalism and the View from Syntax-Semantics*, pp. 161–196, Mouton de Gruyter, Berlin.

Hans-Martin GÄRTNER and Jens MICHAELIS (2010), On the Treatment of Multiple-Wh-Interrogatives in Minimalist Grammars, in Thomas HANNEFORTH and Gisbert FANSELOW, editors, *Language and Logos*, pp. 339–366, Akademie Verlag, Berlin.

Sabrina GERTH (2015), *Memory Limitations in Sentence Comprehension. A Structure-Based Complexity Metric of Processing Difficulty*, Ph.D. thesis, University of Potdsam.

Edward GIBSON (1998), Linguistic Complexity: Locality of Syntactic Dependencies, *Cognition*, 68:1–76.

Edward GIBSON (2000), The Dependency Locality Theory: A Distance-Based Theory of Linguistic Complexity, in Y. MIYASHITA, Alec MARANTZ, and W. O'NEIL, editors, *Image, Language, Brain*, pp. 95–126, MIT Press, Cambridge, MA.

Edward GIBSON and H.-H. Iris WU (2013), Processing Chinese Relative Clauses in Context, *Language and Cognitive Processes*, 28(1-2):125–155.

Peter C. GORDON, Randall HENDRICK, Marcus JOHNSON, and Yoonhyoung LEE (2006), Similarity-Based Interference During Language Comprehension: Evidence From Eye Tracking During Reading, *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 32(6):1304.

Thomas GRAF (2011), Closure Properties of Minimalist Derivation Tree Languages, in Sylvain POGODALLA and Jean-Philippe PROST, editors, *LACL 2011*, volume 6736 of *Lecture Notes in Artificial Intelligence*, pp. 96–111, Springer, Heidelberg, doi:10.1007/978-3-642-22221-4_7.

Thomas GRAF (2012a), Locality and the Complexity of Minimalist Derivation Tree Languages, in Philippe DE GROOT and Mark-Jan NEDERHOF, editors, *Formal Grammar 2010/2011*, volume 7395 of *Lecture Notes in Computer Science*, pp. 208–227, Springer, Heidelberg, doi:10.1007/978-3-642-32024-8_14.

Thomas GRAF (2012b), Movement-Generalized Minimalist Grammars, in Denis BÉCHET and Alexander J. DIKOVSKY, editors, *LACL 2012*, volume 7351 of *Lecture Notes in Computer Science*, pp. 58–73, doi:10.1007/978-3-642-31262-5_4.

Thomas GRAF (2013), *Local and Transderivational Constraints in Syntax and Semantics*, Ph.D. thesis, UCLA.

Thomas GRAF, Alëna AKSËNOVA, and Aniello DE SANTO (2016), A Single Movement Normal Form for Minimalist Grammars, in Annie FORET, Glyn MORRILL, Reinhard MUSKENS, Rainer OSSWALD, and Sylvain POGODALLA, editors, *Formal Grammar: 20th and 21st International Conferences*, pp. 200–215, doi:10.1007/978-3-662-53042-9_12.

Thomas GRAF, Brigitta FODOR, James MONETTE, Gianpaul RACHIELE, Aunika WARREN, and Chong ZHANG (2015), A Refined Notion of Memory Usage for Minimalist Parsing, in *Proceedings of the 14th Meeting on the Mathematics of Language (MoL 2015)*, pp. 1–14, Association for Computational Linguistics, Chicago, USA.

Thomas GRAF and Bradley MARCINEK (2014), Evaluating Evaluation Metrics for Minimalist Parsing, in *Proceedings of the 2014 ACL Workshop on Cognitive Modeling and Computational Linguistics*, pp. 28–36.

John HALE (2001), A Probabilistic Earley Parser as a Psycholinguistic Model, in *Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics*, pp. 1–8.

John HALE (2011), What a Rational Parser Would Do, *Cognitive Science*, 35:399–443.

John T. HALE (2003), *Grammar, Uncertainty and Sentence Processing*, Ph.D. thesis, John Hopkins University.

Henk HARKEMA (2001), A Characterization of Minimalist Languages, in Philippe DE GROOTE, Glyn MORRILL, and Christian RETORÉ, editors, *Logical*

Aspects of Computational Linguistics (LACL'01), volume 2099 of *Lecture Notes in Artificial Intelligence*, pp. 193–211, Springer, Berlin.

Irene HEIM and Angelika KRATZER (1998), *Semantics in Generative Grammar*, Blackwell, Oxford.

Tim HUNTER (2011), Insertion Minimalist Grammars: Eliminating Redundancies between Merge and Move, in Makoto KANAZAWA, András KORNAI, Marcus KRACHT, and Hiroyuki SEKI, editors, *The Mathematics of Language: 12th Biennial Conference*, pp. 90–107, Springer, Berlin, Heidelberg, ISBN 978-3-642-23211-4, doi:10.1007/978-3-642-23211-4_6.

Tim HUNTER (2015a), Deconstructing Merge and Move to Make Room for Adjunction, *Syntax*, 18:266–319, doi:10.1111/synt.12033.

Tim HUNTER (2015b), Left-Corner Parsing of Minimalist Grammars, slides of a talk presented at the First Workshop on Minimalist Parsing, October 10–11 2015, MIT.

Tim HUNTER and Robert FRANK (2014), Eliminating Rightward Movement: Extraposition as Flexible Linearization of Adjuncts, *Linguistic Inquiry*, 45:227–267.

Aravind JOSHI (1990), Processing Crossed and Nested Dependencies: An Automaton Perspective on the Psycholinguistic Results, *Language and Cognitive Processes*, 5:1–27.

Ronald M. KAPLAN (1974), *Transient Processing Load in Relative Clauses*, Ph.D. thesis, Harvard University.

Richard S. KAYNE (1994), *The Antisymmetry of Syntax*, MIT Press, Cambridge, MA.

Edward L. KEENAN and Bernard COMRIE (1977), Noun Phrase Accessiblity and Universal Grammar, *Linguistic Inquiry*, 8:63–99.

John KIMBALL (1973), Seven Principles of Surface Structure Parsing in Natural Language, *Cognition*, 2:15–47.

Gregory M. KOBELE (2006), *Generating Copies: An Investigation into Structural Identity in Language and Grammar*, Ph.D. thesis, UCLA.

Gregory M. KOBELE (2011), Minimalist Tree Languages are Closed Under Intersection with Recognizable Tree Languages, in Sylvain POGODALLA and Jean-Philippe PROST, editors, *LACL 2011*, volume 6736 of *Lecture Notes in Artificial Intelligence*, pp. 129–144, doi:10.1007/978-3-642-22221-4_9.

Gregory M. KOBELE (2015), LF-Copying Without LF, *Lingua*, 166:236–259, doi:10.1016/j.lingua.2014.08.006.

Gregory M. KOBELE, Sabrina GERTH, and John T. HALE (2013), Memory Resource Allocation in Top-Down Minimalist Parsing, in Glyn MORRILL and Mark-Jan NEDERHOF, editors, *Formal Grammar: 17th and 18th International Conferences*, pp. 32–51, doi:10.1007/978-3-642-39998-5_3.

Gregory M. KOBELE, Christian RETORÉ, and Sylvain SALVATI (2007), An Automata-Theoretic Approach to Minimalism, in James ROGERS and Stephan KEPSER, editors, *Model Theoretic Syntax at 10*, pp. 71–80.

Alexander KOLLER and Marco KUHLMANN (2011), A Generalized View on Parsing and Translation, in *Proceedings of the 12th International Conference on Parsing Technologies*, pp. 2–13, Association for Computational Linguistics, Stroudsburg, PA.

Lars KONIECZNY (2005), The Psychological Reality of Local Coherences in Sentence Processing, in *Proceedings of the 27th Annual Conference of the Cognitive Science Society*.

Lars KONIECZNY, Daniel MÜLLER, Wibke HACHMANN, Sarah SCHWARZKOPF, and Sascha WOLFER (2009), Local Syntactic Coherence Interpretation. Evidence from a Visual World Study, in *Proceedings of the 31st Annual Conference of the Cognitive Science Society*, pp. 1133–1138.

Nayoung KWON, Peter C. GORDON, Yoonhyoung LEE, Robert KLUENDER, and Maria POLINSKY (2010), Cognitive and Linguistic Factors Affecting Subject/Object Asymmetry: An Eye-Tracking Study of Prenominal Relative Clauses in Korean, *Language*, 86(3):546–582.

Richard LARSON (1988), On the Double Object Construction, *Linguistic Inquiry*, 19(3):335–391.

David LEBEAUX (1988), *Language Acquisition and the Form of the Grammar*, Ph.D. thesis, University of Massachusetts, Amherst, doi:10.1075/z.97, reprinted in 2000 by John Benjamins.

Roger LEVY (2013), Memory and Surprisal in Human Sentence Comprehension, in Roger P. G. VAN GOMPEL, editor, *Sentence Processing*, pp. 78–114, Psychology Press, Hove.

Chien-Jer Charles LIN and Thomas G. BEVER (2006), Subject Preference in the Processing of Relative Clauses in Chinese, in *Proceedings of the 25th West Coast Conference on Formal Linguistics*, pp. 254–260, Cascadilla Proceedings Project Somerville, MA.

Thomas MAINGUY (2010), A Probabilistic Top-Down Parser for Minimalist Grammars, arXiv:1010.1826v1.

Willem M. MAK, Wietske VONK, and Herbert SCHRIEFERS (2002), The Influence of Animacy on Relative Clause Processing, *Journal of Memory and Language*, 47(1):50–68.

Willem M. MAK, Wietske VONK, and Herbert SCHRIEFERS (2006), Animacy in Processing Relative Clauses: The Hikers That Rocks Crush, *Journal of Memory and Language*, 54(4):466–490.

Axel MECKLINGER, Herbert SCHRIEFERS, Karsten STEINHAUER, and Angela D. FRIEDERICI (1995), Processing Relative Clauses Varying on Syntactic and

Semantic Dimensions: An Analysis with Event-Related Potentials, *Memory & Cognition*, 23(4):477–494.

Jens MICHAELIS (2001), Transforming Linear Context-Free Rewriting Systems into Minimalist Grammars, *Lecture Notes in Artificial Intelligence*, 2099:228–244.

George A. MILLER and Noam CHOMSKY (1963), Finitary Models of Language Users, in R. LUCE, R. BUSH, and E. GALANTER, editors, *Handbook of Mathematical Psychology*, volume 2, pp. 419–491, John Wiley, New York.

George A. MILLER and Kathryn Ojemann MCKEAN (1964), A Chronometric Study of Some Relations Between Sentences, *Quarterly Journal of Experimental Psychology*, 16:297–308.

Edson T. MIYAMOTO and Michiko NAKAMURA (2003), Subject/Object Asymmetries in the Processing of Relative Clauses in Japanese, in *Proceedings of WCCFL*, volume 22, pp. 342–355.

Edson T. MIYAMOTO and Michiko NAKAMURA (2013), Unmet Expectations in the Comprehension of Relative Clauses in Japanese, in *Proceedings of the 35th Annual Meeting of the Cognitive Science Society*, pp. 3074–3079.

Uwe MÖNNICH (2006), Grammar Morphisms, ms. University of Tübingen.

Richard MONTAGUE (1970), English as a Formal Language, in Bruno VISENTINI and ET AL., editors, *Linguaggi nella Società e nella Tecnica*, pp. 189–224, Edizioni di Comunità, Milan.

Frank MORAWIETZ (2003), *Two-Step Approaches to Natural Language Formalisms*, Walter de Gruyter, Berlin, doi:10.1515/9783110197259.

William O'GRADY (2011), Relative Clauses. Processing and Acquisition, in Evan KIDD, editor, *Processing, Typology, and Function*, pp. 13–38, John Benjamins, Amsterdam.

Colin PHILLIPS (1996), *Order and Structure*, Ph.D. thesis, MIT.

Alan PRINCE and Paul SMOLENSKY (2004), *Optimality Theory: Constraint Interaction in Generative Grammar*, Blackwell, Oxford.

Owen RAMBOW and Aravind JOSHI (1995), A Processing Model for Free Word Order Languages, Technical Report IRCS-95-13, University of Pennsylvania.

Philip RESNIK (1992), Left-Corner Parsing and Psychological Plausibility, in *Proceedings of COLING-92*, pp. 191–197.

Sylvain SALVATI (2011), Minimalist Grammars in the Light of Logic, in Sylvain POGODALLA, Myriam QUATRINI, and Christian RETORÉ, editors, *Logic and Grammar — Essays Dedicated to Alain Lecomte on the Occasion of His 60th Birthday*, number 6700 in Lecture Notes in Computer Science, pp. 81–117, Springer, Berlin, doi:10.1007/978-3-642-21490-5_5.

Stuart M. SHIEBER, Yves SCHABES, and Fernando C. PEREIRA (1995), Principles and Implementations of Deductive Parsing, *Journal of Logic Programming*, 24:3–36.

Klaas SIKKEL (1997), *Parsing Schemata*, Texts in Theoretical Computer Science, Springer, Berlin.

Edward P. STABLER (1997), Derivational Minimalism, in Christian RETORÉ, editor, *Logical Aspects of Computational Linguistics*, volume 1328 of *Lecture Notes in Computer Science*, pp. 68–95, Springer, Berlin, doi:10.1007/BFb0052152.

Edward P. STABLER (2006), Sidewards Without Copying, in Gerald PENN, Giorgio SATTA, and Shuly WINTNER, editors, *Formal Grammar '06, Proceedings of the Conference*, pp. 133–146, CSLI, Stanford.

Edward P. STABLER (2011), Computational Perspectives on Minimalism, in Cedric BOECKX, editor, *Oxford Handbook of Linguistic Minimalism*, pp. 617–643, Oxford University Press, Oxford, doi:10.1093/oxfordhb/9780199549368.013.0027.

Edward P. STABLER (2013), Two Models of Minimalist, Incremental Syntactic Analysis, *Topics in Cognitive Science*, 5:611–633, doi:10.1111/tops.12031.

Mark STEEDMAN (2001), *The Syntactic Process*, MIT Press, Cambridge, MA.

Whitney TABOR, Bruno GALANTUCCI, and Daniel RICHARDSON (2004), Effects of Merely Local Syntactic Coherence on Sentence Processing, *Journal of Memory and Language*, 50:355–370.

Shoichi TAKAHASHI and Sarah HULSEY (2009), Wholesale Late Merger: Beyond the A/$\overline{\text{A}}$ Distinction, *Linguistc Inquiry*, 40:387–426.

James W. THATCHER (1967), Characterizing Derivation Trees for Context-Free Grammars Through a Generalization of Finite Automata Theory, *Journal of Computer and System Sciences*, 1:317–322.

Mieko UENO and Susan M GARNSEY (2008), An ERP Study of the Processing of Subject and Object Relative Clauses in Japanese, *Language and Cognitive Processes*, 23(5):646–688.

Jean-Roger VERGNAUD (1974), *French Relative Clauses*, Ph.D. thesis, MIT.

Eric WANNER and Michael MARATSOS (1978), An ATN Approach to Comprehension, in Morris HALLE, Joan BRESNAN, and George A. MILLER, editors, *Linguistic Theory and Psychological Reality*, pp. 119–161, MIT Press, Cambridge, MA.

Jiwon YUN, Zhong CHEN, Tim HUNTER, John WHITMAN, and John HALE (2014), Uncertainty in Processing Relative Clauses Across East Asian Languages, *Journal of East Asian Linguistics*, pp. 1–36.