# Evaluating Evaluation Metrics for Minimalist Parsing

**Thomas Graf**
Department of Linguistics
Stony Brook University
mail@thomasgraf.net

**Bradley Marcinek**
Department of Linguistics
Stony Brook University
bradley.marcinek@stonybrook.edu

## Abstract

In response to Kobele et al. (2012), we evaluate four ways of linking the processing difficulty of sentences to the behavior of the top-down parser for Minimalist grammars developed in Stabler (2012). We investigate the predictions these four metrics make for a number of relative clause constructions, and we conclude that at this point, none of them capture the full range of attested patterns.

## 1 Introduction

Minimalist grammars (MGs; (Stabler, 1997)) are a mildly context-sensitive formalism inspired by Minimalist syntax (Chomsky, 1995), the dominant theory in generative syntax. MGs allow us to evaluate syntactic proposals with respect to computational and cognitive criteria such as generative capacity (Harkema, 2001; Michaelis, 2001) or the memory structures they require (Kobele et al., 2007; Graf, 2012).

A new kind of top-down parser for MGs has recently been presented by Stabler (2011b; 2012). Stabler's parser is noteworthy because it uses derivation trees as a data structure in order to reduce MG parsing to a special case of parsing context-free grammars (CFGs). This raises the question, though, whether derivation trees are a psychologically plausible data structure, and if so, to which extent the Stabler parser makes it possible to test the psycholinguistic predictions of competing syntactic analyses.

In order to address this question, a linking hypothesis is needed that connects the behavior of the parser to a processing difficulty metric. Kobele et al. (2012) — henceforth KGH — propose that the difficulty of sentence $s$ correlates with the maximum number of parse steps the parser has to keep a parse item in memory while processing $s$. This metric is called *maximum tenure* (**Max**). **Max** is appealing because of its simplicity and sensitivity to differences in linguistic analysis, which makes it easy to determine the psycholinguistic predictions of a specific syntactic analyses.

In this paper, we show that **Max** does not make the right predictions for I) relative clauses embedded in a sentential complement and II) subjects gaps versus object gaps in relative clauses. We present a number of simple alternative measures that handle these phenomena correctly, but we also show that these metrics fail in other cases (all results are summarized in Tab. 1 on page 8). We conclude that the prospect of a simple direct link between syntactic analysis and processing difficulty is tempting but not sufficiently developed at this point.

The paper starts with a quick introduction to MGs (Sec. 2.1) and how they are parsed (Sec. 2.2). Section 3 then introduces three alternatives to **Max**. **Max** is then shown to fare worse than those three with respect to well-known contrasts involving relative clauses (Sec. 4). Section 5 briefly looks at three other constructions that pose problems for the alternative metrics.

## 2 Preliminaries

### 2.1 Minimalist Grammars

MGs (Stabler, 1997; Stabler, 2011a) are a highly lexicalized formalism in which structures are built via the operations Merge and Move. Intuitively, Merge enforces local dependencies via subcategorization, whereas Move establishes long-distance filler-gap dependencies.

Every lexical item comes with a non-empty list of unchecked features, and each feature has either positive or negative polarity and is checked by either Merge or Move. Suppose that I) $s$ is a tree whose head has a positive Merge feature $F^+$ as its first unchecked feature, and II) $t$ is a tree whose head has a matching negative Merge feature $F^-$



*Proceedings of the 2014 ACL Workshop on Cognitive Modeling and Computational Linguistics*, pages 28–36,
Baltimore, Maryland USA, June 26 2014. ©2014 Association for Computational Linguistics

as its first unchecked feature. Then Merge checks $F^+$ and $F^-$ and combines $s$ and $t$ into the tree $l(s,t)$ or $l(t,s)$, where $l$ is a label projected by the head of $s$ and $s$ is linearized to the left of $t$ iff $s$ consists of exactly one node. Move, on the other hand, applies to a single tree $s$ whose head $h$ has a positive Move feature $f^+$ as its first unchecked feature. Suppose that $t$ is a subtree of $s$ whose head has the matching negative Move feature $f^-$ as its first unchecked feature. Then Move checks $f^+$ and $f^-$ and returns the tree $l(t,s')$, where $l$ is a label projected by $h$ and $s'$ is obtained by removing $t$ from $s$. Crucially, Move may apply to $s$ iff there is exactly one subtree like $t$. This restriction is known as the Shortest Move Constraint (SMC).

For example, the sentence *John left* involves (at least) the following steps under a simplified Minimalist analysis (Adger, 2003):

$$\text{Merge}(\text{John} :: \text{D}^- \text{ nom}^-, \text{left} :: \text{D}^+ \text{ V}^-) = [_{\text{VP}} \text{ left} :: \text{V}^- \text{ John} :: \text{nom}^- ] \quad (1)$$

$$\text{Merge}(\varepsilon :: \text{V}^+ \text{ nom}^+ \text{ T}^-, (1)) = [_{\text{TP}} \varepsilon :: \text{nom}^+ \text{ T}^- [_{\text{VP}} \text{ left John} :: \text{nom}^- ] ] \quad (2)$$

$$\text{Move}((2)) = [_{\text{TP}} \text{ John } [_{\text{T'}} \varepsilon :: \text{T}^- [_{\text{VP}} \text{ left } ] ] ] \quad (3)$$

This derivation can be represented more succinctly as the *derivation tree* in Fig 1, where all leaves are labeled by lexical items while unary and binary branching nodes are labeled Move and Merge, respectively.

Even though MGs (with the SMC) are weakly equivalent to MCFGs (Michaelis, 2001) and thus mildly context-sensitive in the sense of Joshi (1985), their derivation tree languages can be generated by CFGs (modulo relabeling of interior nodes). As we will see next, this makes it possible to treat MG parsing as a special case of CFG parsing.

### 2.2 Parsing Minimalist Grammars

Thanks to the SMC, the mapping from derivation trees to phrase structure trees is deterministic. Consequently, MG parsing reduces to assigning context-free derivation trees to input sentences, rather than the more complex phrase structure trees. The major difference from CFGs is that the linear order of nodes in an MG derivation tree does not necessarily match the linear order of words in the input sentence — for instance because a moving phrase remains in its base position in the derivation tree. But as long as one can tell for every MG operation how its output is linearized, these discrepancies in linear order can be taken care of in the inference rules of the parser. Stabler (2011b; 2012) shows how exactly this is done for a parser that constructs derivation trees in a top-down fashion. Intuitively, MG top-down parsing is CFG top-down parsing with a slightly different algorithm for traversing/expanding the tree.

Instead of presenting the parser's full set of inference rules, we adopt KGH's index notation to indicate how the parser constructs a given derivation. For instance, if a derivation contains the node ${}^5\text{Merge}_{38}$, this means that the parser makes a prediction at step 5 that Merge occurs at this position in the derivation and keeps this prediction in memory until step 38, at which point the parser replaces it by suitable predictions for the arguments for Merge, i.e. the daughters of the Merge node. Similarly, ${}^{22}\text{the} :: \text{N}^+ \text{ D}^-{}_{28}$ denotes that the parser conjectures this lexical item at step 22 and finally gets to scan it in the input string at step 28.

In principle the parser could simply predict a complete derivation and then scan the input string to see if the two match. In order to obtain an incremental parser, however, scanning steps have to take place as soon as possible. The MG parser implements this as follows: predictions are put into a priority queue, and the prediction with the highest priority is worked on first. The priority of the predictions corresponds to the linear order that holds between the constituents that are obtained from them. For example, if the parser replaces a prediction for a Merge node yielding $l(s,t)$ by predictions $p_s$ and $p_t$ that eventually derive $s$ and $t$, then $p_s$ has higher priority than $p_t$ iff $s$ is predicted to precede $t$. Since Move only takes one argument $s$, replacing a Move prediction by the prediction of $s$ trivially involves no such priority management. However, if movement is to a position to the left of $s$ (as is standard for MGs), none of the lexical items contained within $s$ can be scanned until the entire subtree moving out of $s$ has been predicted and scanned.

If a prediction does not have the highest prior-

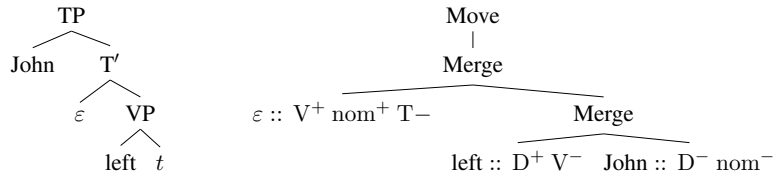


Figure 1: Minimalist phrase structure tree (left) and MG derivation tree (right) for *John left*

ity, it remains in the queue for a few steps before it is expanded into other predictions or discharged by scanning a word from the input string. The number of steps a prediction stays in the queue is called its *tenure*. With KGH's index notation, the tenure of each node is the difference between its indices. Given a parse, its *maximum tenure* **Max** is the smallest $n$ such that the parser stored no prediction in its queue for more than $n$ steps. KGH demonstrate that **Max** can be used to gauge how hard it is for humans to process certain structures. This amounts to equating processing difficulty with memory retention requirements. But as we show in the remainder of this paper, **Max** faces problems with relative clause constructions that were not considered by KGH.

## 3 Alternative Metrics

### 3.1 Three New Metrics

In an attempt to home in on the shortcomings of **Max**, we contrast it with a number of alternative metrics. Since the main advantage of **Max** is its simplicity, which makes it possible to quickly determine the processing predictions of a given syntactic analysis, the metrics we consider are also kept as simple as possible.

**MaxLex** the maximum tenure of all leaves in the derivation

**Box** the maximum number of nodes with tenure strictly greater than 2

**BoxLex** the maximum number of leaves with tenure strictly greater than 2

**MaxLex** is simply the restriction of **Max** to leaf nodes. **Box** and **BoxLex** provide a measure of how many items have to be stored in memory during the parse and hence incur some non-trivial amount of tenure. The threshold is set to 2 rather than 1 to exclude lexical items that are right siblings of another lexical item. In such a case, a single prediction is immediately followed by two consecutive scan steps, which could just as well be thought of as one scan step spanning two words. Nodes with tenure over 2 are highlighted by a box in our derivation trees, hence the name for these two metrics.

All four measures are also divided into two subtypes depending on whether unpronounced leaves (e.g. the empty T-head in Fig. 1) are taken into account — this is inspired by the exclusion of unpronounced material in the TAG-parser of Rambow and Joshi (1995). When reporting the values for the metrics, we thus give slashed values of the form $m/n$, where $m$ is the value with unpronounced leaves and $n$ the value without them.

### 3.2 Methodological Remarks

The following sections investigate the predictions of our difficulty metrics with respect to the embedding of sentential complements versus relative clauses, subject gaps versus object gaps in relative clauses, left embedding, and verb clusters. In order for this comparison to be meaningful, we have to make the same methodological assumptions as KGH.

First, the difficulty metric only has to account for overall sentence difficulty, it does not necessarily correlate with difficulty at a specific word. More importantly, though, all reported processing difficulties are assumed to be due to memory load. This is a very strong assumption. A plethora of alternative accounts are available in the literature. The contrast between subject gaps and object gaps alone has been explained by information-theoretic notions such as surprisal (Hale, 2003; Levy, 2013), the active filler strategy (Frazier and D'Arcais, 1989), or theta role assignment (Pritchett, 1992), to name but a few (see Lin (2006) and Wu (2009) for extensive surveys).

Even those accounts that attribute processing difficulty to memory requirements make ancillary assumptions that are not reflected by the simple memory model entertained here. Gibson's Dependency Locality Theory (1998), for instance, cru-

cially relies on discourse reference as a means for determining how much of a memory burden is incurred by each word.

We take no stance as to whether these accounts are correct. Our primary interest is the feasibility of a memory-based evaluation metric for Stabler's top-down parser. Memory is more likely to play a role in the constructions we look at in the next two sections than in, say, attachment ambiguities or local syntactic coherence effects (Tabor et al., 2004). It may well turn out that memory is not involved at all, but for the purpose of comparing several memory-based metrics, they are the safest starting point.

## 4 Relative Clauses

### 4.1 Empirical Generalizations

Two major properties of relative clauses are firmly established in the literature (see Gibson (1998) and references therein).

- **SC/RC $<$ RC/SC**
  A sentential complement containing a relative clause is easier to process than a relative clause containing a sentential complement.
- **SubjRC $<$ ObjRC**
  A relative clause containing a subject gap is easier to parse than a relative clause containing an object gap.

These generalizations were obtained via self-paced reading experiments and ERP studies with minimal pairs such as (1) and (2), respectively.

(1) 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) 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.

### 4.2 SC/RC and RC/SC

We first consider the contrast between relative clauses embedded inside a sentential complement (SC/RC) and relative clauses containing a sentential complement (SC/RC). Figures 2 and 3 on pages 5 and 6 show the augmented derivations for (1a) and (1b), respectively. For the sake of readability, we omit all features in our derivation trees and instead use standard X′ labels to indicate projection and dashed branches for movement.

Like KGH, we adopt a promotion analysis of relative clauses (Vergnaud, 1974; Kayne, 1994). That is to say, the head noun is selected by an empty determiner to form a DP, which starts out as an argument of the embedded verb and undergoes movement into the specifier of the relative clause (which is treated as an NP). The entire relative clause is then selected by the determiner that would usually select the head noun under the traditional, head-external analysis (Montague, 1970; Chomsky, 1977).[1]

In both derivations the maximum tenure obtains at two points in the matrix clause: I) the unpronounced T-head, and II) the Merge step that introduces the remainder of the VP. The parser must first build the entire subject before it can proceed scanning or expanding material to its right. Consequently, the tenure of these nodes increases with the size of the subject, and since both the SC/RC pattern and the RC/SC pattern necessarily involve large subjects, maximum tenure for both types of sentences is predicted to be relatively high. The parser shows a slightly lower **Max** value for SC/RC than for RC/SC — $32/32$ versus $33/33$.

Although this shows that strictly speaking **Max** is not incompatible with the generalization that SC/RC is easier to process than RC/SC, the difference is so small that even the presence of one more word in the SC/RC sentence could tip the balance towards RC/SC, which seems rather unlikely.

The contrast emerges more clearly with the other measures. **MaxLex** yields the values $32/9$ versus $33/17$, so it fares better than **Max** only if one ignores unpronounced leaves. This is expected since one of the nodes incurring the highest tenure value is the unpronounced T-head. The **Box** values are $14/11$ and $5/3$, and those of **BoxLex** are $12/9$ and $3/1$.

The box values fare better in this case because they are sensitive to the number of dependencies that cannot be discharged immediately. The way the MG parser traverses the tree, a sentential com-

[1] The promotion analysis was chosen to maintain consistency with KGH. But our observations hold for every analysis that involves some movement dependency between the gap and the specifier of the relative clause. This includes the more common head-external analyses mentioned above.

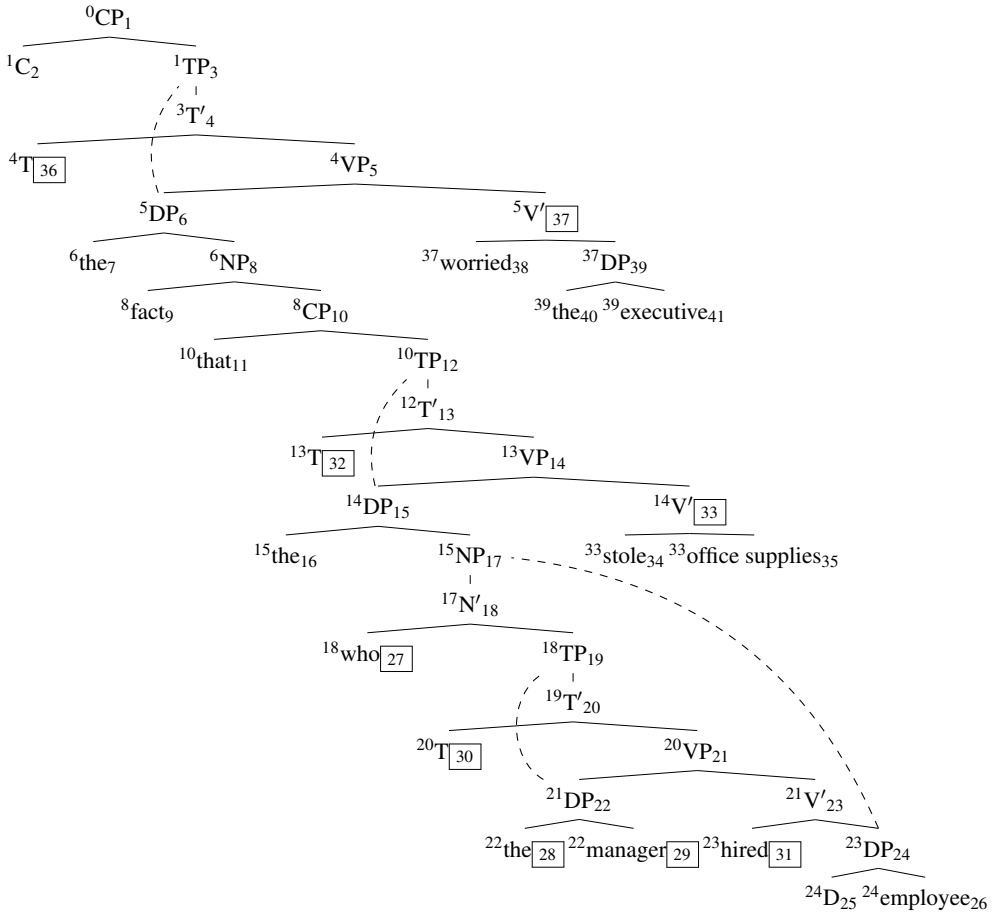


Figure 2: Sentential complement with embedded relative clause; **Max** $= 32/32$, **MaxLex** $= 32/9$, **Box** $= 9/6$, **BoxLex** $= 7/4$

plement in the subject position of a relative clause cannot be fully processed until the movement dependency within the relative clause has been taken care of. So even though the sentential complement is explored first, all its predicted elements must be kept in memory. A relative clause within the subject of a sentential complement, on the other hand, poses less of a challenge because the movement of its containing subject is so short that it only delays the processing of the T-head and V′.

### 4.3 Subject Gaps and Object Gaps

A stronger argument against **Max** is furnished by the preference for subject gaps over object gaps: maximum tenure is always the same for both constructions. Consider the derivations in Fig. 4 and 5 on pages 7 and 8. They have the same **Max** value because the maximum tenure once again obtains at the T-head of the matrix clause and the Merge node that expands the matrix VP. The tenure of these nodes is determined by the size of the subject, which contains the relative clause. But since the size of the subject is not affected by whether it is the subject or the object that is extracted from the relative clause, maximum tenure will never vary between these two constructions.

Once again the alternative metrics fare better than **Max**. **MaxLex** evaluates to $19/7$ and $19/9$. As before the tenure on the T-head causes **MaxLex** to behave like **Max** unless unpronounced words are ignored. If one does so, however, the maximum tenure value occurs on the relative pronoun *who* instead. Since *who* is the head of the relative clause, it is introduced early on during the structure building process, but it cannot be scanned until the parser reaches the element that moves into its specifier. Objects are more deeply embedded than subjects, and consequently it takes the parser less time to reach the subject than the object. As a result, *who* has greater tenure if the relative clause contains an object gap instead of a subject gap.

**Box** and **BoxLex** also predict the attested con-

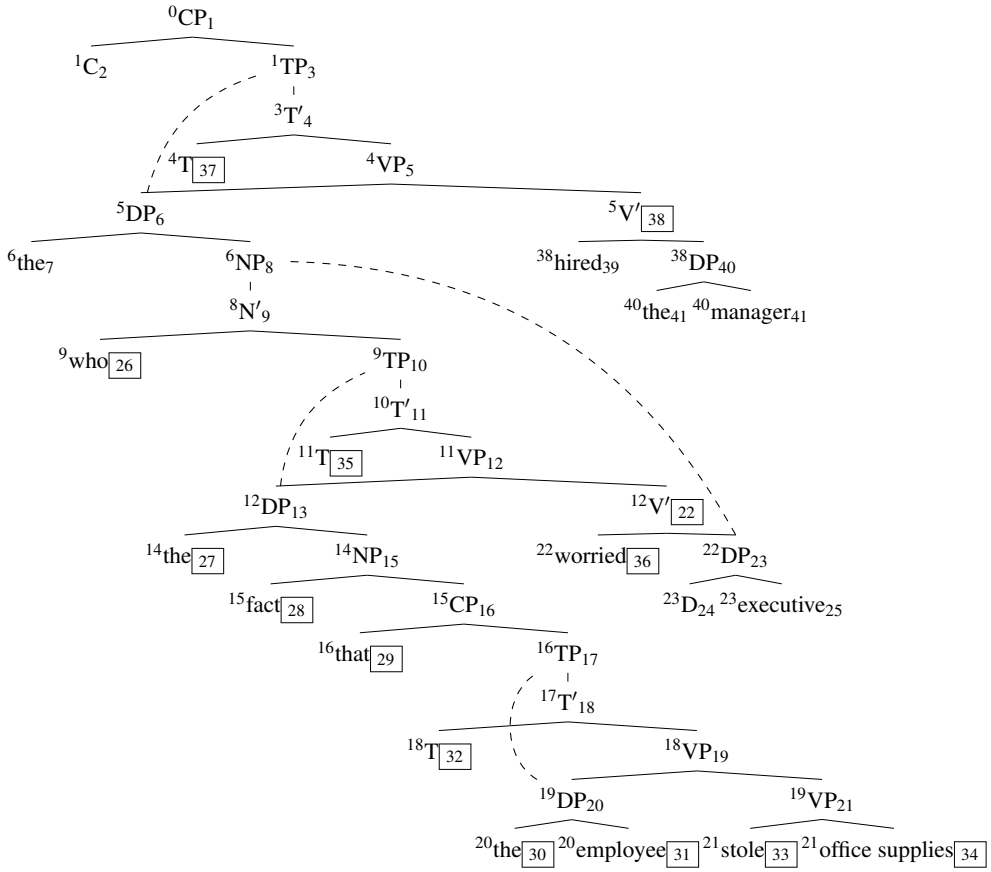


Figure 3: Relative clause containing a sentential complement; **Max** $= 33/33$, **MaxLex** $= 33/17$, **Box** $= 14/11$, **BoxLex** $= 12/9$

trast. **Box** produces the values $5/3$ and $7/5$, whereas **BoxLex** returns $3/1$ and $6/4$. Since subjects are introduced at a higher position than objects, movement of the subject causes fewer nodes to be delayed in their processing — the VP has not been fully expanded yet, so the nodes contained by it do not need to be stored in memory because the parser hasn't even predicted them at this point.

## 5 Further Observations

### 5.1 Verb Clusters in Dutch and German

KGH show that **Max** correctly predicts the attested difficulty differences between German and Dutch verb clusters (Bach et al., 1986). German verb clusters instantiate nested dependencies of the form $\text{DP}_1\ \text{DP}_2 \cdots \text{DP}_n\ \text{V}_n \cdots \text{V}_2\ \text{V}_1$. Dutch verb clusters, on the other hand, show crossing dependencies: $\text{DP}_1\ \text{DP}_2 \cdots \text{DP}_n\ \text{V}_1\ \text{V}_2 \cdots \text{V}_n$. Even though the latter not context-free and hence computationally more complex than the former, they are actually easier to process. Since KGH's account relies on the tenure of (pronounced) leaves, it also carries over to **MaxLex**.[2]

**Box** and **BoxLex**, however, do not make this prediction. In both Dutch and German every $\text{V}_i$ has to be kept in memory before it can be scanned, so that a sentence with $n$ verbs will have $n$ boxes. According to **Box** and **BoxLex**, there should be no processing difference between German and Dutch. This can be partially fixed by summing the tenure of all boxed nodes so that overall memory load is at least partially taken into account, yielding the measures **SumBox** and **SumBoxLex**. But even those still make the wrong prediction for $n < 4$, that is to say, they establish the desired difference only after a point where both cluster types are already very hard to process.

[2] Strictly speaking KGH build their argument on the tenure of T, which **MaxLex** must ignore for the constructions investigated in this paper. However, tenure can be measured at $\text{V}_1$ instead, in which case Dutch clusters with three or more verbs have lower **MaxLex** values than the corresponding German clusters. Clusters consisting of only two verbs have the same **MaxLex** value in both languages. An anonymous reviewer points out that this is exactly the pattern found by Bach et al. (1986).

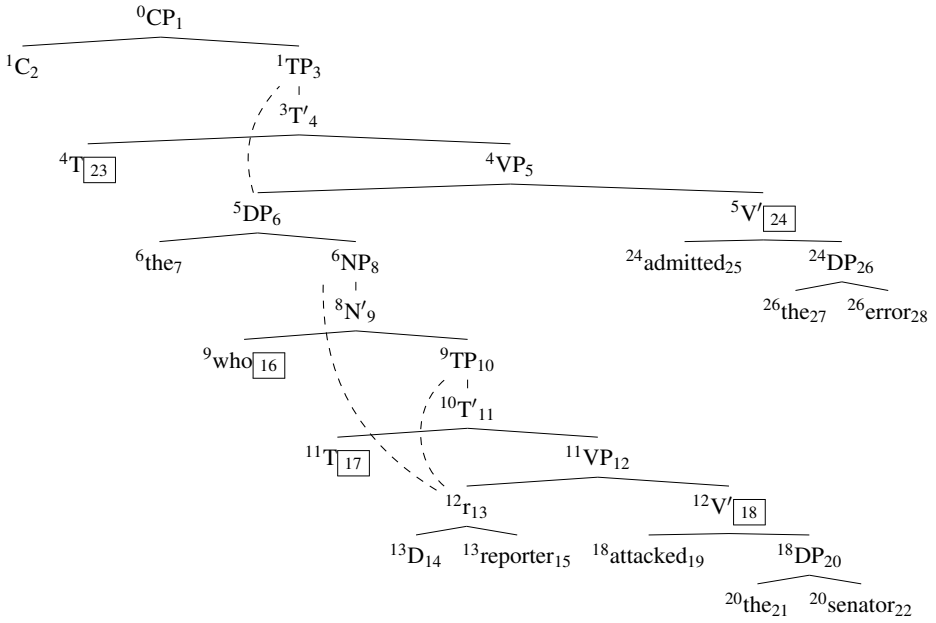


Figure 4: Relative clause with subject gap; **Max** $= 19/19$, **MaxLex** $= 19/7$, **Box** $= 5/3$, **BoxLex** $= 3/1$

### 5.2 Left Embedding

KGH note that if processing difficulty is determined by **Max**, then left embedding constructions such as English possessor nesting should lead to a sharp increase in parsing difficulty similar to center-embedding, which is not the case (Resnik, 1992).

(3) [[[Mike ['s uncle]] ['s cousin]] ['s roommate]] went to the store.

**Box** makes a similar prediction, whereas **MaxLex** and **BoxLex** do not (cf. Tab. 1 on page 1). Keep in mind that a left embedding construction $c$ increases the tenure of the right sibling of $c$ with every level of embedding. As long as $c$ is not a lexical item, it will be ignored by **MaxLex** and **BoxLex**. Therefore possessor-embedding is predicted to be unproblematic, whereas a right-adjunction structure as in [$_{VP}$ [$_{VP}$ [$_{VP}$ left ] quickly ] yesterday ] should increase the processing load. While we are not aware of any studies on this topic, such a split strikes us as highly unnatural.

### 5.3 Head-Final Relative Clauses

Preliminary work of ours suggests that almost none of the metrics covered in this paper work for languages where relative clauses precede their head nouns, such as Chinese, Japanese, and Korean. There is overwhelming evidence that these languages still show a preference for subject gaps over object gaps (Lin, 2006; Wu, 2009; Kwon et al., 2013). The syntactic structure of relative clauses in these languages is up to debate; but assuming that they involve rightward movement of the head noun into a specifier of the relative clause followed by remnant leftward movement of the TP into another specifier, most metrics derive a preference for object gaps (see the last two rows in Tab. 1). Only **Box** shows a small advantage for subject gaps.

## 6 Discussion and Future Work

Several metrics were compared in this paper that measure processing difficulty in terms of very different parameters: I) how long an item stays in memory (**Max**, **MaxLex**), II) how many items must be stored in memory (**Box**, **BoxLex**), and III) for what kind of material these criteria matter ($\pm$lexical, $\pm$pronounced).

A quick glance at Tab. 1 reveals that no clear winner emerges. **Box** and **BoxLex** fail to capture the differences between Dutch and German verb clusters, whereas **Max** struggles with relative clause constructions and left embedding. **MaxLex** captures all these fact if only pronounced elements are taken into account, but makes the dubious prediction that right adjunction of a single word should be harder than left embedding or right adjunction of an adjunct that consists of at least two words. In addition, **MaxLex** fails to derive a subject gap preference for head-final relative clauses.

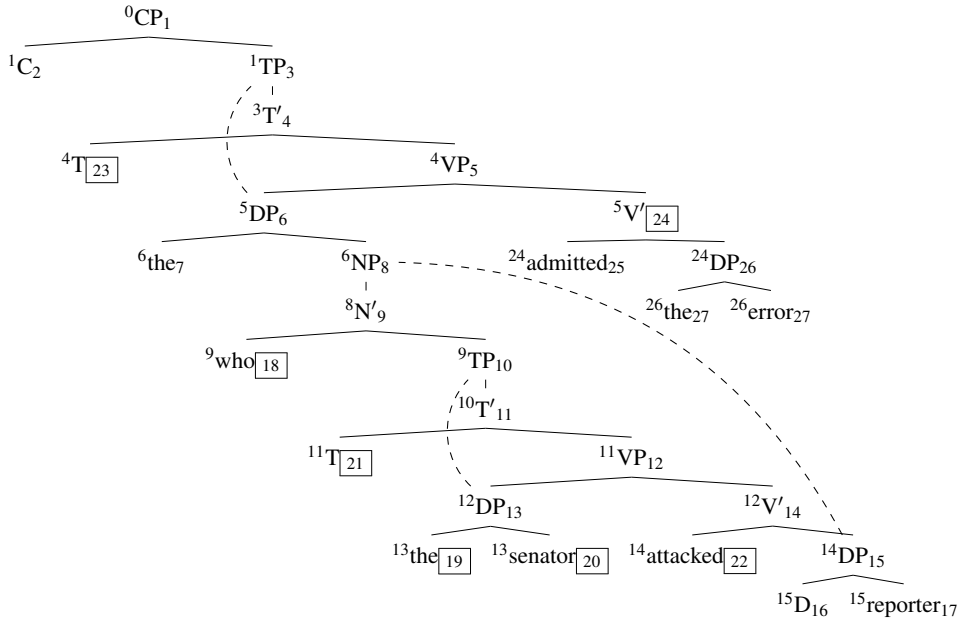


Figure 5: Relative clause with object gap; **Max** $= 19/19$, **MaxLex** $= 19/9$, **Box** $= 7/5$, **BoxLex** $= 6/4$

| **Phenomenon** | **Max** | **MaxLex** | **Box** | **BoxLex** | **SumBox** | **SumBoxLex** |
|---|---|---|---|---|---|---|
| SC/RC | 32/32 | 32/9 | 9/6 | 7/4 | 142/81 | 91/30 |
| RC/SC | 33/33 | 33/17 | 14/11 | 12/9 | 219/149 | 186/116 |
| subject gap RC | 19/19 | 19/7 | 5/3 | 3/1 | 57/32 | 32/7 |
| object gap RC | 19/19 | 19/9 | 7/5 | 6/4 | 78/49 | 59/30 |
| 1 possessor | 7/7 | 7/2 | 2/1 | 1/0 | 14/7 | 7/0 |
| 2 possessors | 11/11 | 11/2 | 3/2 | 1/0 | 27/16 | 11/0 |
| 3 possessors | 15/15 | 15/2 | 4/3 | 1/0 | 46/31 | 15/0 |
| 1 right adjunct | 7/7 | 7/3 | 3/2 | 2/1 | 17/10 | 10/3 |
| 2 right adjuncts | 12/12 | 12/8 | 5/4 | 4/3 | 42/30 | 30/18 |
| 3 right adjuncts | 15/15 | 15/12 | 7/6 | 6/5 | 58/43 | 43/28 |
| crossing $<$ nesting | yes | yes | no | no | partially | partially |
| head-final subj RC | 20/20 | 20/11 | 5/4 | 4/3 | 66/39 | 46/19 |
| head-final obj RC | 20/20 | 20/10 | 6/4 | 3/1 | 63/38 | 35/10 |

Table 1: Overview of evaluation metrics

It is very likely that a more complicated metric could account for all these facts. But the appeal of **Max** and the alternatives investigated here is their simplicity. A simple metric is easier to study from a formal perspective. In an ideal world, the metric would turn out to correlate with a basic tree-geometric property of derivations so that the processing predictions of syntactic analyses can be determined at a glance without simulations or large-scale corpus work.

Two routes seems promising at this point. In order to rule out that the problem isn't with the metrics but rather the MG parser itself, the metrics should be tested with other parsing models. Those need not even be based on MGs, since the metrics measure aspects of memory management, which is an integral part of every parser.

Alternatively, we may look into how the metrics are applied. An anonymous reviewer points out that **Max** derives the preference for subject gaps if derivations that tie for **Max** are then compared with respect to the second-highest tenure value, which is $7/7$ for subject gaps and $10/9$ for object gaps. While this still leaves us with cases like left embedding where **Max** predicts a higher processing load than expected, it eliminates the problem of **Max** incorrectly equating two structures.

## Acknowledgments

We are extremely grateful to the three anonymous reviewers. Their extensive comments not only brought about many improvements in the presentation and structure of the paper but also made us consider the wider implications of the work reported here.

## References


David Adger. 2003. *Core Syntax: A Minimalist Approach*. Oxford University Press, Oxford.

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.

Noam Chomsky. 1977. *Essays on Form and Interpretation*. New York, North Holland.

Noam Chomsky. 1995. *The Minimalist Program*. MIT Press, Cambridge, Mass.

Lyn Frazier and Flores D'Arcais. 1989. Filler driven parsing: A study of gap filling in Dutch. *Journal of Memory and Language*, 28:331–344.

Edward Gibson. 1998. Linguistic complexity: Locality of syntactic dependencies. *Cognition*, 68:1–76.

Thomas Graf. 2012. 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*, pages 208–227, Heidelberg. Springer.

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*, pages 193–211. Springer, Berlin.

Aravind Joshi. 1985. Tree-adjoining grammars: How much context sensitivity is required to provide reasonable structural descriptions? In David Dowty, Lauri Karttunen, and Arnold Zwicky, editors, *Natural Language Parsing*, pages 206–250. Cambridge University Press, Cambridge.

Richard S. Kayne. 1994. *The Antisymmetry of Syntax*. MIT Press, Cambridge, Mass.

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*, pages 71–80.

Gregory M. Kobele, Sabrina Gerth, and John T. Hale. 2012. Memory resource allocation in top-down minimalist parsing. In *Proceedings of Formal Grammar 2012*.

Nayoung Kwon, Robert Kluender, Marta Kutas, and Maria Polinsky. 2013. Subject/object processing asymmetries in korean relative clauses: Evidence from ERP data. *Language*, 89:537–585.

Roger Levy. 2013. Memory and surprisal in human sentence comprehension. In Roger P. G. van Gompel, editor, *Sentence Processing*, pages 78–114. Psychology Press, Hove.

Chien-Jer Charles Lin. 2006. *Grammar and Parsing: A Typological Investigation of Relative-Clause Processing*. Ph.D. thesis, University of Arizona.

Jens Michaelis. 2001. Transforming linear context-free rewriting systems into minimalist grammars. *Lecture Notes in Artificial Intelligence*, 2099:228–244.

Richard Montague. 1970. English as a formal language. In Bruno Visentini and et al., editors, *Linguaggi nella Societ e nella Tecnica*, pages 189–224. Edizioni di Comunit, Milan.

Bradley L. Pritchett. 1992. *Grammatical Competence and Parsing Performance*. University of Chicago Press, Chicago.

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*, pages 191–197.

Edward P. Stabler. 1997. Derivational minimalism. In Christian Retoré, editor, *Logical Aspects of Computational Linguistics*, volume 1328 of *Lecture Notes in Computer Science*, pages 68–95. Springer, Berlin.

Edward P. Stabler. 2011a. Computational perspectives on minimalism. In Cedric Boeckx, editor, *Oxford Handbook of Linguistic Minimalism*, pages 617–643. Oxford University Press, Oxford.

Edward P. Stabler. 2011b. Top-down recognizers for MCFGs and MGs. In *Proceedings of the 2011 Workshop on Cognitive Modeling and Computational Linguistics*. to appear.

Edward P. Stabler. 2012. Bayesian, minimalist, incremental syntactic analysis. *Topics in Cognitive Science*, 5:611–633.

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.

Jean-Roger Vergnaud. 1974. *French Relative Clauses*. Ph.D. thesis, MIT.

Fuyun Wu. 2009. *Factors Affecting Relative Clause Processing in Mandarin*. Ph.D. thesis, University of Southern California.