Wide-Coverage Neural A* Parsing for Minimalist Grammars

John Torr Miloš Stanojević Mark Steedman Shay B. Cohen

School of Informatics
University of Edinburgh
11 Crichton Street, Edinburgh, UK
john.torr@cantab.net m.stanojevic@ed.ac.uk
steedman@inf.ed.ac.uk scohen@inf.ed.ac.uk

Abstract

Minimalist Grammars (Stabler, 1997) are a computationally oriented, and rigorous formalisation of many aspects of Chomsky's (1995) Minimalist Program. This paper presents the first ever application of this formalism to the task of realistic wide-coverage parsing. The parser uses a linguistically expressive yet highly constrained grammar, together with an adaptation of the A* search algorithm currently used in CCG parsing (Lewis and Steedman, 2014; Lewis et al., 2016), with supertag probabilities provided by a bi-LSTM neural network supertagger trained on MGbank, a corpus of MG derivation trees. We report on some promising initial experimental results for overall dependency recovery as well as on the recovery of certain unbounded long distance dependencies. Finally, although like other MG parsers, ours has a high order polynomial worst case time complexity, we show that in practice its expected time complexity is $\mathcal{O}(n^3)$. The parser is publicly available.[1]

1 Introduction

Parsers based on linguistically expressive formalisms, such as Head-Driven Phrase Structure Grammar (HPSG; Pollard and Sag 1994) and Combinatory Categorial Grammar (CCG; Steedman 1996), were shown in Rimell et al. (2009) and Nivre et al. (2010) to be more effective at recovering certain unbounded long-distance dependencies than those merely approximating human grammar with finite state or context-free covers. Such dependencies can be vital for tasks like open domain question answering, for example. Furthermore, as proven independently by Huybregts (1984) and Shieber (1985), some languages exhibit constructions which put them beyond even the weak generative capacity of any context-free grammar. The investigation of parsing systems based on more powerful (mildly) context-sensitive formalisms has therefore been a very active area of research within the field of computational psycholinguistics over the past 35 years (see, e.g., Joshi 1985, 1990; Rambow and Joshi 1994; Steedman 2000; Hale 2011; Stabler 2013; Stanojević and Stabler 2018).

Another linguistically expressive grammatical framework is Transformational Grammar (Chomsky, 1957, 1965, 1981), the latest incarnation of which is the Minimalist Program (MP; Chomsky 1995). A defining property of MP is that constituents move. For example, in 1a below, *what* moves to the left periphery of the matrix clause from a deep subject position and will therefore be interpreted as the semantic AGENT of *eat*; in 1b, meanwhile, it moves from the deep object position and so is interpreted instead as the semantic PATIENT of *eat*.

(1) a. What$_i$ do you think t$_i$ eats mice?
b. What$_i$ do you think mice eat t$_i$?

MP continues to dominate much of theoretical syntax, and Stabler's (1997) rigorous formalisation of this framework has proven a popular choice for investigations into human sentence processing (Hale, 2003; Kobele et al., 2013; Stabler, 2013; Graf and Marcinek, 2014; Graf et al., 2015; Gerth, 2015; Stanojević and Stabler, 2018). On the other hand, TG has enjoyed far less popularity within computational linguistics more generally,[2] which is unfortunate given that it is arguably the most extensively developed syntactic theory across the greatest number of languages, many of which are otherwise under-resourced. Conversely, the process of constructing large grammar fragments and

[1] https://github.com/mgparsing/astar_mg_parser

[2] For an anti-Chomskyan perspective on why this disconnect came about, see Pullum (2009).

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 2486–2505
Florence, Italy, July 28 - August 2, 2019.

subjecting these to computational testing can have a salutary impact on the syntactic theory itself, forcing choices between competing analyses of the same construction, and exposing incompatibilities between analyses of different constructions, along with areas of over/undergeneration which may otherwise go unnoticed (Bierwisch 1963; Abney 1996; both cited in Müller 2016).

The received wisdom within NLP is that TG/MP is too complex and insufficiently formalised to be applied to realistic parsing tasks (see Müller 2016 for discussion). Such assumptions prompted Sproat and Lappin (2005) to issue a challenge to the Minimalist community which has hitherto gone unanswered: to construct a wide-coverage statistical parser trained in a supervised fashion and exhibiting performance that is comparable with other state-of-the-art parsers. This paper is the first to take up this challenge, and will introduce the first ever wide-coverage parser in the Minimalist (and arguably the entire TG) tradition, along with some promising initial experimental results. The parser is equipped with a linguistically expressive, wide-coverage grammar based on an extended version of Stabler's (1997) Minimalist Grammars (MG) formalism, which is a rigorously formal, computationally oriented and polynomially parseable interpretation of mainstream MP that is weakly equivalent to Multiple Context-Free Grammars (MCFG; Seki et al. 1991). The parser itself is an adaptation of a highly efficient A* CCG parsing algorithm (Lewis and Steedman, 2014) with a bi-LSTM model trained on MGbank, an MG version of the English Penn Treebank (PTB; Marcus et al. 1993) currently under development.

2 Background

Beginning in the 1960s, a number of parsers were developed which implemented aspects of the various iterations of Chomskyan syntactic theory (e.g. Petrick 1965; Zwicky et al. 1965; Woods 1970, 1973; Plath 1973; Marcus 1980; Kuhns 1990; Fong 1991; Stabler 1992; Fong and Ginsburg 2012), but most of these systems operated over relatively closed domains and were never evaluated against wide-coverage treebank test data.

Principar (Lin, 1993), and its descendant Minipar (Lin, 1998, 2001), are the only truly wide-coverage parsers in the Chomskyan tradition of which we are aware. Minipar incorporates MP's bare phrase structure and some of its economy principles. It is also statistical, having been self-trained on a 1GB corpus. However, while these parsers model the phrase structure and locality constraints of TG, they are not transformational: movement is merely 'simulat[ed]' (Lin, 1993, page 116) by passing features up a precompiled network of nodes representing a tree, from the site of the trace to the site of the antecedent, with the latter merged directly into its surface position, in the style of GPSG. Furthermore, in this approach, antecedents necessarily c-command their traces (Lin, 1993, page 115), presumably making these parsers unsuitable for implementing MP analyses involving *remnant movement* (see Stabler 1999).

2.1 MG parsers

A number of parsers have been developed for Stablerian MGs, which do allow for actual movement, including remnant movement. What all working MG parsers (Harkema, 2001; Hale, 2003; Stabler, 2013; Stanojević and Stabler, 2018) have until now shared in common is that they are small-scale theoretical implementations equipped only with toy lexicons/grammars. There has been a limited amount of research into probabilistic MGs, notably in generative locally normalised models (Hale, 2003; Hunter and Dyer, 2013). However, these works remain so far untested owing to the unavailability, until very recently, of any MG treebank for training and evaluating models.

2.2 MGbank

MGbank (Torr, 2017, 2018) is a treebank of MG derivation trees constructed in part manually by hand-annotating a subset of PTB sentences and in part automatically using a parser equipped with the manually constructed grammar and guided by the corresponding PTB and CCGbank (Hockenmaier and Steedman, 2007) structures. The corpus was continuously machine tested for over- and undergeneration throughout its development. It currently covers over 463,000 words of the PTB, or nearly 56% of its trees, and contains over 47,100 lexical entries and over 1,100 MG lexical categories. The average sentence length in MGbank is 16.9 (vs 21.7 in the PTB) and the maximum sentence length is 50. The derivation trees produced by the parser have also been transduced into Xbar and MG derived phrase structure trees.

The MGbank grammar has been designed to capture many long distance dependencies not included in the original treebank, including the bind-

ing of reflexive/reciprocal anaphors and floating quantifiers by their antecedents, the dependency between the two subconstituents of a discontinuous quoted expression (*"funny thing," says the kicker, "both these candidates are named Rudolph Giuliani."*), the licensing of polarity items such as *anything*, *anymore* and *much* by interrogative and negation heads (*you have *(not) eaten anything*), and the distributional dependency between expletive *there* and its obligatorily indefinite DP associate (*there seem to be some/several/*the/*those problems*). All of these long distance dependencies, along with those involved in control, raising, topicalization and wh movement, are integrated into the grammar itself, obviating the need for separate post-processing techniques to recover them (Johnson, 2002; Cahill et al., 2004). The MG lexical categories have also been annotated with over 100 fine-grained selectional and agreement restriction features (e.g. +3SG, -NOM, +INF, MASC, +INDEF, +FOR, MNR, +LOC, etc) to avoid many instances of unwanted overgeneration.

Movement is clearly a very powerful operation. However, it is constrained here using many of the locality constraints proposed in the TG literature. These include not only Stabler's (1997) strict version of the Shortest Move Constraint, but also a partially derelativized version (DSMC) inspired by Rizzi (1990), along with versions of the specifier/adjunct island constraints, the right roof constraint, complex NP constraint, coordinate structure constraint, *that*-trace filter, Principle A of Chomsky's (1981) Binding Theory, and so on.

3 Minimalist Grammars

Our parser uses the MG formalism described in Torr and Stabler (2016; henceforth T&S) and Torr (2018, 2019). Here we give only a brief overview. MGs are strongly lexicalised, with ordered feature sequences on lexical categories determining both the subcategorization frames of words and the movement operations which must apply. There are four basic types of structure building features: =x/x= selectors and x selectees, and +f licensors and -f licensees. Selectors and selectees trigger Merge operations, with x= indicating rightward selection and =x leftward selection (similar to the forward and backward slash notation in CCG). Licensors and licensees trigger Move operations. Except for a single c selectee at the root of the tree, all features entering the derivation must be checked and deleted by applying one of a small set of (here, around 45) abstract binary Merge and unary Move rules; these rules concatenate and reorder expressions' string components.

Consider the following MG lexicon.

ϵ, they, ϵ :: d
ϵ, saw, ϵ :: d= =d v
ϵ, who, ϵ :: d -wh
ϵ, [int], ϵ :: v= +WH c

Each entry consists of a string component, followed by a type separator,[3] followed by a sequence of syntactic features. The epsilons represent empty strings and are slots for left and right dependent strings to be merged into.[4] Strings enclosed in square brackets are also empty, and appear in this form at the lexical level only simply to make the trees easier to read. Figure 1 shows the MG derivation tree for the embedded question *who they saw*, along with its corresponding phrase structure tree in which λ indicates an empty node position from which a phrase has moved (informally, a trace); the leaf nodes of the derivation tree are lexical items while the final surface string appears at the root node; binary nodes represent Merge operations while unary nodes represent Move operations. The interesting step occurs at the lowest binary node: because *who* has a -wh licensee still to check, its string is not merged into the right ϵ (complement) slot of *saw* when these two items are Merged; instead, it is kept in a separate moving *chain* until its -wh feature is checked by the +WH of [int] via an application of Move.

4 The Parser

Our parser uses an adaptation of the A* search algorithm for CCG presented in Lewis and Steedman (2014) (henceforth, L&S). In this section we first review that algorithm, before going on to show how it was adapted to the MG formalism.

4.1 A* CCG parsing

Combinatory Categorial Grammar (CCG; Steedman 2000) is another linguistically expressive formalism capable of recovering unbounded long distance dependencies. Like MG, CCG is strongly lexicalised, with a large lexical category set and a

[3] :: is used for lexical items, and : for derived items.

[4] Heads are kept separate from their left and right dependents to allow for head movement operations (Stabler, 2001)

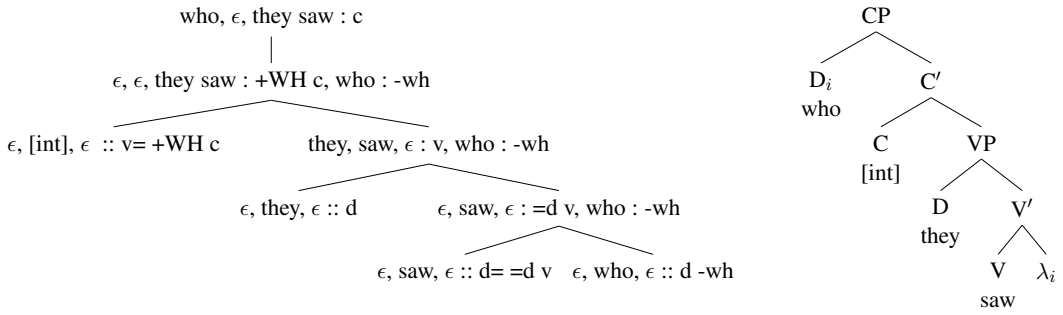

Figure 1: MG derivation tree (left) and phrase structure tree (right) for the embedded question *who they saw*. The derivation has been simplified for ease of exposition by removing case and head movements, as well as the null tense and light verb heads.

small set of abstract combinatory rules, the most basic of which is forward/backward application (equivalent to MG's Merge). Categories are either basic (NP, S, etc) or functional. The functional categories determine the subcategorization frame of the words they label. For example, the category for a transitive verb is (S\NP)/NP, which says that this word must combine with an (object) NP on its right (indicated by the forward slash), which will yield a category which must combine with a second (subject) NP on its left (indicated by the backward slash). In place of movement, CCG uses type raising and function composition rules to capture unbounded long distance dependencies.

CCG already has a very well-established research tradition in wide-coverage parsing (see, e.g., Hockenmaier and Steedman 2002; Clark and Curran 2007b; Lewis and Steedman 2014; Xu 2016; Lewis et al. 2016; Wu et al. 2017). A key advancement in CCG parsing that enabled it to become efficient enough to support large-scale NLP tasks was the introduction of Markovian supertagging techniques in Clark and Curran (2007b) that were borrowed from Lexicalised Tree Adjoining Grammar (LTAG; Bangalore and Joshi 1999). Supertagging is essentially just part-of-speech tagging for strongly lexicalised formalisms, which have much larger tagsets than the 50 or so tags used in the PTB. Because the supertags predetermine much of the combinatorics, this is sometimes referred to as 'almost parsing'.

Inspired by the A* algorithm for PCFGs of Klein and Manning (2003), L&S present a simple yet highly effective CCG parsing model which is factored over the probabilities assigned by the lexical supertagger alone, with no explicit model of the derivation at all. This approach is highly efficient and avoids the need for aggressively pruning the search space, which degraded the performance of earlier CKY CCG parsers. Instead, the parser considers the complete distribution of the 425 most commonly occurring CCG lexical categories for each word. The supertagger was originally a unigram log-linear classifier, but Lewis et al. (2016) greatly enhanced its accuracy by exchanging this for a stacked bi-LSTM neural model.

The key difference between A* and CKY CCG parsing is the fact that A* uses search heuristics that avoid building the whole chart without compromising the correctness guarantees. This is achieved using an agenda implemented as a priority queue of items ranked by their *cost*, calculated as a product of their *inside cost* and an upper bound on their expected *outside cost*. The agenda is initialised with the full set of 425 supertags for each word. The parser pops the item with the lowest cost from the agenda, stores it in the chart if it is not already there, and attempts to combine it with other items already present the chart. Newly created items have their costs calculated before being added to the priority queue agenda. The entire process is repeated until a complete parse for the sentence is returned. The algorithm guarantees that the first parse returned is the most probable (i.e. the Viterbi parse) according to the model.

L&S treat a CCG parse y as a list of lexical categories $c_0 \ldots c_{n-1}$ together with a derivation, and make the simplifying assumptions that all derivations licensed by the grammar are equally likely, and that the probability of a given lexical category assignment is conditionally independent of all the other assignments given the sentence. Let $\mathcal{Y}$ be the set of all derivations licensed by the grammar; then the optimal parse $\hat{y}$ for a given sentence S with words $w_0 \ldots w_{n-1}$ is given as:

$$\hat{y} = \text{argmax}_{y \in \mathcal{Y}} \prod_{i=0}^{n-1} p(c_i \mid S) \qquad (1)$$

Let α be a set of indices {i,..,j} for words $w_i...w_j$ labelled with category sequence $c_i...c_j$ inside some expression. The inside probability of α is simply the product of the probabilities of the lexical category assignments given the sentence.

$$s(\alpha) = \prod_{i \in \alpha} p(c_i \mid S) \tag{2}$$

The upper bound estimate for the outside probability of a span α is given by

$$h(\alpha) = \prod_{i \notin \alpha} \max_{c_i} p(c_i \mid S) \tag{3}$$

where $\max_{c_i} p(c_i \mid S)$ is the probability of the most likely category assigned to word w_i according to the supertagger, which can be precomputed for the sentence and cached. To avoid numerical errors caused by multiplying together very small numbers, we convert the probabilities to log space *costs* and use addition rather than multiplication.

4.2 A* MG parsing

The simplicity, speed and performance of L&S's A* CCG parser made it attractive for a first implementation of a wide-coverage MG parser. However, while CCG and MG are similar in some respects[5] (such as the fact that they are both strongly lexicalised), there are also some fundamental differences between the formalisms which mean that some adaptations are needed in order to port this A* algorithm to MGs. The first (trivial) issue is that MG derivations feature discontinuous spans in order to allow for movement, as we saw in Figure 1. Therefore, we must redefine α in Equations 2 and 3 to be the set of word indices covered by *all* the spans contained within an MG expression.

The second issue is that, following T&S, the MGbank grammar allows for so-called Across-the-Board (ATB) head and phrasal movements in order to capture adjunct control, parasitic gaps, and certain coordination structures. ATB phrasal movement is illustrated in 2 below.

(2) Who$_i$ did Jack say Mary likes t_i and Pete hates t_i?

In 2, *who* has moved from two separate base generated object positions in across-the-board fashion. T&S (adapting ideas in Kobele 2008) propose to account for this by initially generating two instances of *who* in the two object positions and then later unifying them into a single item when the second conjunct is merged into the main structure. For A*, when two expressions containing unifiable movers are merged together, only one of those movers must contribute to the cost of the resulting expression in order to avoid excessive penalisation for what is now just a single instance of the moving item. We can achieve this for both ATB head and phrasal movement by first calculating the sum of the costs of the two expressions that are Merged, and then subtracting from this the cost of one member of each pair of unified movers.

In the MGbank grammar (unlike in Kobele 2008), it can be the case that two unified (head) movers have different derivational histories, in which case they may well have different costs[6]. In such cases, the parser uses the greater of these two costs when calculating the inside cost of the newly formed expression. If the lower of the two costs were used instead, it may make some costs non-monotonically increasing.[7]

The final problem relates to the fact that, unlike CCG, MG allows for phonetically null heads (following mainstream MP), but supertaggers can only tag the overt words of a sentence. However, we would like our probability model to also be defined over the null heads. Addressing this problem, Torr (2018) proposes an algorithm for extracting a set of complex LTAG-like MG lexical supertag categories from a corpus of MG derivation trees, which we adopt here. Each supertag contains precisely one overt atomic MG lexical item and zero or more atomic null heads anchored to it. For example, in Figure 1, the [int] head would be included inside the supertag anchored by *saw*. The supertagging model can now be refactored over these complex, overt MG categories; the parser continues to manipulate the atomic categories, but now keeps track of the fact that the v= of [int] must obligatorily be checked by the v feature of (this specific instance of) *saw*, and vice versa. During parsing, the overt heads carry the entire cost of their supertag into the agenda; the null heads are simply assigned a zero cost.

Pseudocode for the A* MG parser can be found in Appendix A.

[5] See Berwick and Epstein (1995) on the convergence of Minimalist syntax and Categorial Grammar.

[6] Another difference is that T&S do not adopt the GPSG-style slash feature mechanism used in Kobele (2008).

[7] Note that one drawback to only using the cost of one of the two unified instances is that the strict optimality guarantees of A* are lost.

5 Experiments

5.1 Model description

We used two types of MG grammars in our experiments: Abstract and Reified. The difference between them is that in the Abstract grammar, most of the 100 or so fine-grained selectional and agreement restriction features have been removed with the exception of the following 5 features, which are necessary to the inner workings of the parser: ANA, EDGE, IT, +NONE, MAIN. The Reified grammar is clearly more constrained, which should make it more precise (at some expense to recall) but at the same time more difficult to supertag correctly due to the sparsity that comes with a higher number of supertags. Extracting the complex MG supertags from the entire MGbank corpus resulted in a Reified tagset of 3926 items and an Abstract tagset of 2644 items.[8]

For both Abstract and Reified we used the same supertagging neural architecture that works by initially embedding the word tokens using the final layer of an ELMo embedder (Peters et al., 2018), followed by a single affine transformation to compress the embeddings into a vector of size 128 for each word. These embeddings are further fed into a two layer bi-LSTM (Hochreiter and Schmidhuber, 1997; Graves, 2013). Finally, the hidden states of the final layer of the bi-LSTM are passed through a two layer MLP to predict the distribution of the supertags for each word. The parameters are trained using an Adam optimizer with a learning rate of 0.0002.

5.2 Recovering MGBank dependencies

We first tested the parser on its ability to recover global syntactic and semantic (local and non-local) dependencies extracted from MGbank. We extracted labelled and unlabelled bi-lexical dependencies for each binary non-terminal in the Xbar phrase structure trees transduced from the MG derivation trees.[9] To make up for the shortfall in the number of trees in MGbank, we used both sections 00 and 01 for development and both sections 23 and 24 for testing, with sections 02-22 used for training.

		model	F1	P	R	E
syntax	LAB	Abstract	79.33	81.87	76.94	21.01
		Reified	**80.10**	**83.43**	**77.02**	**21.61**
	ULAB	Abstract	84.57	87.15	**82.14**	29.59
		Reified	**85.19**	**88.63**	82.02	**30.49**
semantics	LAB	Abstract	74.90	77.17	**72.75**	20.96
		Reified	**75.47**	**78.53**	72.64	**21.56**
	ULAB	Abstract	83.69	86.16	**81.36**	33.30
		Reified	**84.11**	**87.47**	81.01	**34.50**

Table 1: Results on the whole MGbank test set with P, R and E indicating precision, recall and exact match respectively.

Table 1 shows the results on the MGbank test set. On both dependency types, the Reified model has higher precision, F1-score and exact matching, but has a lower score on recall owing to the constraining impact of the selectional and agreement features: The Abstract model returned parses for 1924 sentences out of 1998 in the test set (i.e. 96.5%), while the Reified model returned 1902 (i.e. 95.4%). The F1 scores in table 1 are respectable for a first attempt at wide-coverage MG parsing, though it should be noted that the MGbank test set is somewhat easier than the PTB test set owing to the difference of 4.8 in average sentence length between the two corpora.

5.3 Comparison to CCG

Cross-formalism comparison is in general a difficult task (Clark and Curran, 2007a) because it is necessary to account both for (1) the differences in how the parsers work and (2) the differences in the kinds of structures they predict. To control for (1) we re-implemented a CCG parser similar to L&S's CCG A* algorithm but using our supertagger to make the comparison fair. We first trained our CCG supertagger on the CCG trees from CCGbank, but only on those sentences that are also present in MGbank. We then tested the CCG parser on the recovery of CCGbank dependencies for the test sentences also appearing in

[8] This number of tags is closer to the 4727 elementary trees of the TAG treebank of Chen (2001) than to CCGbank's (Hockenmaier and Steedman, 2007) 1286 lexical categories.

[9] As in Collins (1999), the labels are triples of the parent, non-head child and head child categories. The dependencies include both local dependencies and those created by movement, hence this evaluation is more akin to the deep dependency evaluation discussed in Clark et al. (2002) for CCG than to the more standard practice of evaluating parsers in terms of just local dependencies (e.g. Collins 1999). The semantic head of the clause is taken to be the main verb, while its syntactic head, if present, is the overt complementizer; similarly, nouns are taken to be semantic heads of PPs and DPs while their syntactic heads are the preposition and determiner respectively; the semantic heads of coordination structures are the conjuncts themselves, while the syntactic head is the coordinator. Unlabelled dependencies are also undirected, as is standard practice in CCG evaluation.

	model	F1	P	R	E
LAB	Our CCG A*	**87.4**	87.2	**87.6**	**40.0**
	EasyCCG A*	83.8	87.2	80.7	31.4
ULAB	Our CCG A*	**92.8**	92.5	**93.0**	**47.2**
	EasyCCG A*	90.1	**93.8**	86.8	35.9

Table 2: Results of CCG parsers on all 1994 sentences of MGbank test set for CCG dependencies.

MGbank, and compared this to an off the shelf CCG parser, namely EasyCCG, that was trained over the whole of the CCGbank training set. The results are shown in Table 2. Our CCG parser shows much better performance in spite of being trained on much less data than EasyCCG, making it a tough point of comparison for our MG parser.

To account for (2) we compared the CCG and MG parsers on their ability to recall the dependencies for which both CCGbank and MGbank agree by taking as the test set the intersection of the gold unlabelled undirected CCGbank and syntactic MGbank dependencies for sentences appearing in the MGbank test set. Precision cannot be computed due to the difficulties in normalising predictions on the CCG and MG sides: one might predict more dependencies which may be correct but are not predicted by the syntactic theory used in the other parser and therefore would be penalised.

The results of this evaluation are shown in Table 4. The CCG parser clearly exhibits superior performance, although the MG parser performs respectably given that it is up against a near state-of-the-art parser for a formalism with a much longer history in wide-coverage parsing. The higher performance of the CCG parser is likely the result of a more complete search due to the lower complexity of the formalism (the CCG parser parsed all sentences) and of the much smaller supertag set that is easier to predict as evident in Table 3. This means that the MG parser requires a larger amount of training data than the CCG parser to achieve similar levels of accuracy and efficiency (because the speed of A* parsing depends on the quality of the probabilistic model). We tried replacing all MG supertags occurring less than twice in the training data with UNK tags to reduce the noise from unreliable tags, but this hurt performance. Once MGbank's coverage is increased, the difference between the formalisms may narrow.

The MG parser is a prototype Python implementation, and to keep parsing times practical the search space was pruned so that only the 40 most likely supertags per word were retained. Even so, the parser still timed out on a few sentences in the test set. Once reimplemented in a faster language, its recall should increase as it will have more time to explore a less aggressively pruned search space.

top k	CCG	MG Abstract	MG Reified
1	95.73	83.11	80.62
5	99.41	97.22	95.89
10	99.64	98.42	97.66
20	99.78	99.01	98.42
40	99.83	99.26	98.81

Table 3: Supertagging accuracies for each grammar as the probability of having the correct supertag in the top-k predictions per word.

parser	R	E
CCG A*	95.30	69.03
MG Abstract A*	91.75	54.38
MG Reified A*	92.65	55.67

Table 4: Results on overlapping gold CCGbank and syntactic MGbank dependencies in sections 23 and 24.

5.4 Parsing speed

The CKY MG parser of Harkema (2001), when augmented with head movement, has a worst case time complexity of $\mathcal{O}(n^{4k+12})$ where k is the maximum number of phrasal movers that can be contained in any single expression. In the MGbank formalism, owing to DSMC, $k = 4$ (see Torr 2019), meaning that the worst case complexity of parsing with this formalism using Harkema's algorithm would be $\mathcal{O}(n^{28})$. Our A* parsing algorithm operates in a similar fashion, except that it takes an additional multiplicative cost of $\mathcal{O}(\log n)$ due to the usage of a heap data structure for implementing the agenda. $\mathcal{O}(n^{28} \log n)$ is, of course, a prohibitively high time complexity. However, although A* does not improve on the worst case theoretical complexity of CKY, it can dramatically improve its practical expected complexity.

Figure 2 shows the scatter plot of parsing times for different sentence lengths and the average curve. The average curve is less informative in very long sentences due to the smaller number of parses, but in regions where there are more data points a clear pattern can be observed: a cubic polynomial curve approximates average time taken to parse sentences extremely well, which means that the expected time complexity of MG

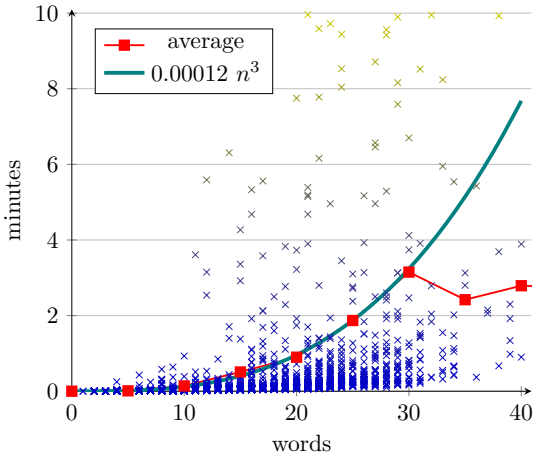

Figure 2: Parsing speed for Abstract model on test set.

parsing with our grammar and statistical model is $\mathcal{O}(n^3)$. This is much better than the worst case analysis, although the variance is high, with some sentences still requiring a very long time to parse.

Recently, Stanojević (2019) has shown that with relatively small adjustments to the parser's inference rules, MGs with head movement can be parsed in $\mathcal{O}(n^{2k+5})$ time in the worst case,[10] which for the MGbank grammar equates to $\mathcal{O}(n^{13})$, a dramatic improvement over $\mathcal{O}(n^{28})$. We hope to leverage these efficiency gains in the future to improve the expected time complexity of the parser.

5.5 Coverage

Section 00 of the PTB contains 1921 sentences with an average sentence length of 21.9 words; other than a 212 word outlier, the maximum sentence length is 96. When run over all of these sentences, the Reified parser returned parses for 1490 (77.6%) sentences with an average sentence length of 14 and a maximum sentence length of 53. The Abstract parser returned 1549 parses (80.6%) with an average sentence length of 15.3 and a maximum sentence length of 49. The CCG A* parser returned 1909 parses (99.4%).

5.6 Recovery of unbounded dependencies

As noted in Section 1, the recovery of unbounded dependencies, including wh-object questions, is a primary motivation for using linguistically expressive parsers in NLP. Wh-object questions themselves are extremely rare in the PTB, but object relative clauses, which also involve unbounded movement, are relatively frequent. Following Clark et al. (2004), we manually evaluated our parser on the free and non-free object (and embedded subject) relative clauses in section 00 of the PTB, as well as on the two examples of so-called *tough movement*. The MGbank analyses of these constructions are discussed in Appendix B.

There are 24 examples of non-free object relative dependencies across 20 sentences in section 00, and 17 free object relative dependencies across 16 sentences. All of these sentences, along with indications of which dependencies our parser did and did not recover, are given in Appendix C, and are presented using the MGbank tokenization used by the MG A* parser (the CCG A* parser used the original CCGbank tokenization).

On the free object relatives, our Abstract parser performed best, recovering 13/17 dependencies. The parser only predicted 14 free object relatives meaning that the precision was 13/14. Of the 4 free object relative dependencies in the data which it missed, 3 were in very long sentences on which the parser timed out (the time-out was set to 30 mins), suggesting that a faster re-implementation may achieve higher recall. In the one case which the parser actually got wrong, it correctly identified that there was a free object relative dependency, but extracted the wrong object from a double object verb. Clark et al. (2004) reported recall of 14/17 (with precision 14/15), while our A* CCG parser recovered 15.5/17 of the free object relative dependencies with precision also 15.5/17.

Non-free object relatives are harder than both wh object questions and free object relatives because they require a head noun to be identified in addition to an extraction site. Our Abstract parser performed best here, retrieving 10/24; the CCG A* parser recovered 15/24, with precision of 15/21 (Clark et al. (2004) also reported recall of 15/24 and precision of 15/20). Our Reified parser retrieved 13/24 with precision 13/17 when allowed to reparse any sentences it initially failed to find any analyses for with increasingly relaxed tag-dictionary settings. In two of the errors, the parser correctly identified the extraction site, but attached the relative clause to the wrong NP. For example, in sentence 1, the parser attached *whom*

[10] Fowlie and Koller (2017) previously demonstrated that MGs *without head movement* could be parsed in $\mathcal{O}(n^{2k+3})$ worst case time, which was already a dramatic improvement over Harkema's original result. However, Stanojević (2019) shows that adding head movement to Fowlie and Koller's system increases complexity to $\mathcal{O}(n^{2k+9})$.

Sony hosted for a year to *complaint* rather than to *American*. Appositive relative clauses such as this are treated as involving adjunction of the relative clause to the head noun in MGbank, and the choice of attachment to either *American* or *complaint* is underdetermined by the model (the same supertag containing the requisite [rel] and [adjunctizer] heads will be assigned to *hosted* in either case).[11] For the restrictive relative clause in sentence 8, the parser incorrectly assigned the supertag containing the [relativizer] null head (which causes the noun to undergo *promotion*) to the noun *esteem* rather than to *damage*, hence the problem here originates with the scores assigned by the supertagger. In the other two errors, the parser incorrectly predicted an object extraction dependency, again owing to tagging mistakes.

We also evaluated on the 2 tough movement examples in section 00, one of which is shown below.

(3) That$_i^A$ got hard [$_{CP}$ t$_i^{A'}$ to take t$_i^A$].

Tough movement is of linguistic interest because it arguably involves a DP licensed in two case positions as well as so-called *improper movement*, in which an A′-movement step feeds subsequent A-movement. In order to generate tough movements, MGbank uses a null [op] head which has the effect of a unary type-changing rule mapping an ordinary DP into a DP with additional A- and A′-movement licensees.

Our parser failed to correctly analyse either of the two examples in section 00 owing to supertagging errors. For example, in 3 there are three important tagging decisions to be made: *hard* must be assigned the supertag for a tough adjective, *that* the supertag for a pronoun which undergoes tough movement,[12] and *take* the supertag for a transitive verb. The highest scoring tag assigned to *hard* by the Abstract supertagger was the supertag for a regular adjective that takes a CP complement (*eager to help*). The correct tough adjective supertag, meanwhile only ranked 14th, meaning that the A* search algorithm never got to consider it. Furthermore, the highest ranked tag for *take* was the supertag for an unergative intransitive verb; the correct transitive verb tag appeared in second place. Finally, the supertag for a pronoun undergoing tough movement was not included in the 40 tags assigned to *that* owing to the fact that this supertag did not appear in the training data at all. We tried increasing the 8 examples of tough movement in the training data to 18 examples (including one example with *that* as the tough mover) by performing some additional hand annotation of PTB sentences. This bolstered the tough adjective supertag to 10th position, while the tough movement supertag for *that* now appeared in 28th position, but this was not enough to enable the parser to correctly recover the tough movement analysis.

Our A* CCG parser scored 1/2 (the same as Clark et al. 2004); its higher performance is no doubt due to the much smaller tag set and the fact that CCG does not require special supertags for tough-moved DPs.

6 Conclusion

We have presented the first ever wide-coverage Transformational Grammar parser. The results of this initial attempt are optimistic. First, the accuracy on recovering syntactic and semantic dependencies predicted by the Minimalist syntax is reasonable considering the higher complexity of the mechanisms behind Minimalism compared to other formalisms. In comparison to CCG, a formalism with a much longer history of wide-coverage parsing, performance currently lags behind. However, the gap will likely narrow as the size and quality of MGbank improves and as better probabilistic models are developed enabling these systems to parse a higher number of sentences.

Another important and optimistic result of this investigation is that Minimalist Grammar parsing is not as slow as may have been expected given its worst case time complexity. Worst case complexity results are sometimes raised as a criticism of TG theories. Our results show that the combination of a good neural probabilistic model and A* search, together with a strong formal grammar, makes Minimalist parsing practical for the majority of sentences.

Acknowledgments

The first author was supported by an EPSRC PhD studentship, the second author by an ERC H2020 Advanced Fellowship GA 742137 SEMANTAX grant, the third author by a Google Faculty Award, and the fourth author by a Bloomberg award. We would also like to thank the three anonymous reviewers for their helpful feedback.

[11] One way to resolve such ties would be to augment the supertag-factored model with a head-dependency model.

[12] This supertag contains both the overt category assigned to *that* and the [op] null head (see Figure 6) in Appendix B.

References

Steven Abney. 1996. Statistical methods and linguistics. In *The Balancing Act: Combining Symbolic and Statistical Approaches to Language*, pages 1–26. MIT Press.

Srinivas Bangalore and Aravind Joshi. 1999. Supertagging: An approach to almost parsing. *Computational Linguistics*, 25:237–265.

Robert C Berwick and Samuel D Epstein. 1995. Computational minimalism: The convergence of the minimalist syntactic program and categorial grammar. *TWLT-10: Algebraic Methods in Language Processing, Enschede, the Netherlands*.

Rajesh Bhatt. 2002. The raising analysis of relative clauses: Evidence from adjectival modification. *Natural language semantics*, 10(1):43–90.

Manfred Bierwisch. 1963. *Grammatik des deutschen Verbs*. Akademie Verlag.

Michael Brody. 1993. θ-theory and arguments. *Linguistic Inquiry*, pages 1–23.

A Cahill, M Burke, R O'Donovan, J van Genabith, and A Way. 2004. Long-distance dependency resolution in automatically acquired wide-coverage pcfg-based lfg approximations. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, pages 320–327, Barcelona, Spain. Association for Computational Linguistics.

John Chen. 2001. *Towards efficient statistical parsing using lexicalized grammatical information*. Ph.D. thesis, University of Delaware.

Noam Chomsky. 1957. *Syntactic Structures*. Mouton, The Hague.

Noam Chomsky. 1965. *Aspects of the Theory of Syntax*. MIT Press, Cambridge, MA.

Noam Chomsky. 1977. On wh-movement. *Formal syntax*, pages 71–132.

Noam Chomsky. 1981. *Lectures on Government and Binding*. Foris, Dordrecht.

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

Noam Chomsky and Howard Lasnik. 1977. Filters and control. *Linguistic inquiry*, 8(3):425–504.

Stephen Clark and James Curran. 2007a. Formalism-independent parser evaluation with ccg and depbank. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 248–255. Association for Computational Linguistics.

Stephen Clark and James R. Curran. 2007b. Wide-coverage efficient statistical parsing with CCG and log-linear models. *Computational Linguistics*, 33:493–552.

Stephen Clark, Julia Hockenmaier, and Mark Steedman. 2002. Building deep dependency structures with a wide-coverage ccg parser. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 327–334. Association for Computational Linguistics.

Stephen Clark, Mark Steedman, and James R Curran. 2004. Object-extraction and question-parsing using ccg. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Michael Collins. 1999. *Head-Driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania.

Sandiway Fong. 1991. The computational implementation of principle-based parsers. In Robert Berwick, Steve Abney, and Carol Tenny, editors, *Principle-Based Parsing*, pages 65–82. Kluwer, Dordrecht.

Sandiway Fong and Jason Ginsburg. 2012. Computation with doubling constituents: Pronouns and antecedents in phase theory. In Anna Maria Di Sciullo, editor, *Towards a Biolinguistic understanding of grammar: Essays on interfaces*, pages 303–338. John Benjamins.

Meaghan Fowlie and Alexander Koller. 2017. Parsing minimalist languages with interpreted regular tree grammars. In *Proceedings of the Thirteenth International Workshop on Tree Adjoining Grammar and Related Formalisms (TAG+13)*, pages 11–20. Association for Computational Linguistics.

Michael L Fredman and Robert Endre Tarjan. 1987. Fibonacci heaps and their uses in improved network optimization algorithms. *Journal of the ACM (JACM)*, 34(3):596–615.

Sabrina Gerth. 2015. *Memory limitations in sentence comprehension*. Ph.D. thesis, University of Potsdam.

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)*, pages 1–14. Association for Computational Linguistics.

Thomas Graf and Bradley Marcinek. 2014. Evaluating evaluation metrics for minimalist parsing. In *Proceedings of the Fifth Workshop on Cognitive Modeling and Computational Linguistics*, pages 28–36.

Alex Graves. 2013. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*.

John Hale. 2003. *Grammar, Uncertainty and Sentence Processing*. Ph.D. thesis, Johns Hopkins University.

John T Hale. 2011. What a rational parser would do. *Cognitive Science*, 35(3):399–443.

Hendrik Harkema. 2001. *Parsing Minimalist Languages*. Ph.D. thesis, UCLA, Los Angeles, California.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Julia Hockenmaier and Mark Steedman. 2002. Generative models for statistical parsing with combinatory categorial grammar. In *In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 335–342. Association for Computational Linguistics.

Julia Hockenmaier and Mark Steedman. 2007. Ccgbank: A corpus of ccg derivations and dependency structures extracted from the penn treebank. *Computational Linguistics*, 33(3):355–396.

Norbert Hornstein. 2001. *Move! A Minimalist Theory of Construal*. Blackwell Publishing.

Tim Hunter and Chris Dyer. 2013. Distributions on minimalist grammar derivations. In *Proceedings of the 13th Meeting on the Mathematics of Language (MoL 13)*, pages 1–11, Sofia, Bulgaria. The Association of Computational Linguistics.

Riny Huybregts. 1984. The weak inadequacy of context-free phrase-structure grammars. In Ger de Haan, Mieke Trommelen, and Wim Zonneveld, editors, *Van Periferie naar Kern*, pages 81–99. Foris, Dordrecht.

Mark Johnson. 2002. A simple pattern-matching algorithm for recovering empty nodes and their antecedents. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 136–143, Philadelphia. Association for Computational Linguistics.

Aravind Joshi. 1985. Tree-adjoining grammars. In David Dowty, Lauri Karttunen, and Arnold Zwicky, editors, *Natural Language Parsing*, pages 206–250. Cambridge University Press, Cambridge.

Aravind K Joshi. 1990. Processing crossed and nested dependencies: An automation perspective on the psycholinguistic results. *Language and cognitive processes*, 5(1):1–27.

Richard S. Kayne. 1994. *The Antisymmetry of Syntax, Linguistic Inquiry Monograph Twenty-Five.* MIT Press, Cambridge, Massachusetts.

Dan Klein and Christopher D Manning. 2003. A* parsing: fast exact viterbi parse selection. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 40–47. Association for Computational Linguistics.

Gregory M Kobele. 2008. Across-the-Board Extraction in Minimalist Grammars. In *Proceedings of the Ninth International Workshop on Tree Adjoining Grammar and Related Formalisms (TAG+9)*, volume 9, pages 113–128. Association for Computational Linguistics.

Gregory M Kobele, Sabrina Gerth, and John Hale. 2013. Memory resource allocation in top-down minimalist parsing. In *International Conference on Formal Grammar*, pages 32–51. Springer, Association for Computational Linguistics.

Robert J. Kuhns. 1990. A PARLOG implementation of government-binding theory. In *13th International Conference on Computational Linguistics, COLING 1990, University of Helsinki, Finland, August 20-25, 1990*, pages 394–396.

Mike Lewis, Kenton Lee, and Luke Zettlemoyer. 2016. Lstm CCG parsing. In *Proceedings of the 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 221–231. Association for Computational Linguistics.

Mike Lewis and Mark Steedman. 2014. A* ccg parsing with a supertag-factored model. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 990–1000.

Dekang Lin. 1993. Principle-based parsing without overgeneration. In *Proceedings of the 31st annual meeting on Association for Computational Linguistics*, pages 112–120. Association for Computational Linguistics.

Dekang Lin. 1998. Dependency-based evaluation of minipar. In *Proceedings of the Workshop on the Evaluation of Parsing Systems, First International Conference on Language Resources and Evaluation*, Granada, Spain.

Dekang Lin. 2001. Latat: Language and text analysis tools. In *Proceedings of the first international conference on Human language technology research*, pages 1–6. Association for Computational Linguistics.

Wolfgang Maier, Miriam Kaeshammer, and Laura Kallmeyer. 2012. Plcfrs parsing revisited: Restricting the fan-out to two. In *Proceedings of the 11th International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG+ 11)*, pages 126–134. Association for Computational Linguistics.

Mitch Marcus, Beatrice Santorini, and M. Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19:313–330.

Mitchell P Marcus. 1980. *Theory of syntactic recognition for natural languages*. MIT press.

Stefan Müller. 2016. *Grammatical theory: From transformational grammar to constraint-based approaches*. Language Science Press.

Mark-Jan Nederhof. 2003. Weighted deductive parsing and knuth's algorithm. *Computational Linguistics*, 29(1):135–143.

Joakim Nivre, Laura Rimell, Ryan McDonald, and Carlos Gomez-Rodriguez. 2010. Evaluation of dependency parsers on unbounded dependencies. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 833–841. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Stanley Roy Petrick. 1965. *A recognition procedure for transformational grammars.* Ph.D. thesis, Massachusetts Institute of Technology.

Warren J Plath. 1973. Transformational grammar and transformational parsing in the request system. In *COLING 1973 Volume 2: Computational And Mathematical Linguistics: Proceedings of the International Conference on Computational Linguistics*, volume 2.

Carl Pollard and Ivan Sag. 1994. *Head Driven Phrase Structure Grammar*. CSLI Publications, Stanford, CA.

Geoffrey Pullum. 2009. Computational linguistics and generative linguistics: The triumph of hope over experience. In *Proceedings of the EACL 2009 Workshop on the Interaction between Linguistics and Computational Linguistics*, pages 12–21, Athens, Greece. Lawrence Erlbaum Associates.

Andrew Radford. 2004. *Minimalist Syntax: Exploring the Structure of English*. Cambridge University Press.

Owen Rambow and Aravind K. Joshi. 1994. A processing model for free word order languages. In C. Clifton Jr., L. Frazier, and K. Rayner, editors, *Perspectives on Sentence Processing*. L. Erlbaum, Hillsdale, NJ.

Laura Rimell, Stephen Clark, and Mark Steedman. 2009. Unbounded dependency recovery for parser evaluation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 813–821, Singapore. Association for Computational Linguistics.

Luigi Rizzi. 1990. *Relativized minimality.* The MIT Press.

Hiroyuki Seki, Takashi Matsumura, Mamoru Fujii, and Tadao Kasami. 1991. On multiple context free grammars. *Theoretical Computer Science*, 88:191–229.

Stuart Shieber. 1985. Evidence against the context-freeness of natural language. *Linguistics and Philosophy*, 8:333–343.

Richard Sproat and Shalom Lappin. 2005. A challenge to the Minimalist community. *Linguist List*, 16:1156.

Edward P. Stabler. 1992. *The logical approach to syntax: foundations, specifications, and implementations of theories of government and binding*. MIT Press.

Edward P. Stabler. 1997. Derivational minimalism. In *Logical Aspects of Computational Linguistics (LACL'96)*, volume 1328 of *Lecture Notes in Computer Science*, pages 68–95, New York. Springer.

Edward P. Stabler. 1999. Remnant movement and complexity. In Gosse Bouma, Erhard Hinrichs, Geert-Jan M. Kruijff, and Richard Oehrle, editors, *Constraints and resources in natural language syntax and semantics*, volume 2, pages 299–326. CSLI Stanford, CA.

Edward P. Stabler. 2001. Recognizing head movement. In *Logical Aspects of Computational Linguistics: 4th International Conference, LACL 2001, Le Croisic, France, June 27-29, 2001, Proceedings.*, volume 4, pages 245–260.

Edward P. Stabler. 2013. Two models of minimalist, incremental syntactic analysis. *Topics in Cognitive Science*, 5:611–633.

Miloš Stanojević and Edward Stabler. 2018. A sound and complete left-corner parsing for minimalist grammars. In *Proceedings of the Eight Workshop on Cognitive Aspects of Computational Language Learning and Processing*, pages 65–74.

Miloš Stanojević. 2019. On the computational complexity of head movement and affix hopping. In *Formal Grammar 2019*. Springer Berlin Heidelberg.

Mark Steedman. 1996. *Surface Structure and Interpretation*. Linguistic Inquiry Monograph 30. MIT Press, Cambridge, MA.

Mark Steedman. 2000. *The Syntactic Process*. MIT Press, Cambridge, Massachusetts.

John Torr. 2017. Autobank: a semi-automatic annotation tool for developing deep minimalist grammar treebanks. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, Software Demonstrations, Valencia, Spain, April 3-7 2017*, pages 81–86. Association for Computational Linguistics.

John Torr. 2018. Constraining mgbank: Agreement, l-selection and supertagging in minimalist grammars. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 590–600. Association for Computational Linguistics.

John Torr. 2019. *Wide-Coverage Statistical Parsing with Minimalist Grammars*. Ph.D. thesis, University of Edinburgh.

John Torr and Edward P. Stabler. 2016. Coordination in minimalist grammars: Excorporation and across the board (head) movement. In *Proceedings of the Twelfth International Workshop on Tree Adjoining Grammar and Related Formalisms (TAG+12)*, pages 1–17. Association for Computational Linguistics.

William A Woods. 1970. Transition network grammars for natural language analysis. *Communications of the ACM*, 13(10):591–606.

William A Woods. 1973. An experimental parsing system for transition network grammars. *Natural language processing*, pages 111–154.

Huijia Wu, Jiajun Zhang, and Chengqing Zong. 2017. A dynamic window neural network for ccg supertagging. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)*, pages 3337–3343.

Wenduan Xu. 2016. Lstm shift-reduce ccg parsing. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1754–1764. Association for Computational Linguistics.

Arnold M Zwicky, Joyce Friedman, Barbara C Hall, and Donald E Walker. 1965. The mitre syntactic analysis procedure for transformational grammars. In *Proceedings of the November 30–December 1, 1965, fall joint computer conference, part I*, pages 317–326. ACM.

A Appendix: Pseudocode for the MG A* algorithm

Algorithm 1 MG parser A* algorithm

```
1:  while agenda is not empty do
2:      item1 ← deleteMax(agenda)
3:      if item1 is goal item then
4:          return item1
5:      else if item1 ∉ chart then
6:          add(chart, item1)
7:          R ← [ ]
8:          if can move item1 then
9:              add(R, move(item1))
10:         for item2 ∈ chart do
11:             if can merge item1 and item2 then
12:                 add(R, merge(item1, item2))
13:         for item ∈ R do
14:             if item ∉ {chart ∪ agenda} then
15:                 add(agenda, item)
16:             else if item ∈ agenda then
17:                 updateWeight(agenda, item)
```

The A* search algorithm presented in Algorithm 1 is an adaptation of the weighted deductive parsing approach (Nederhof, 2003; Maier et al., 2012) to Minimalist Grammars. It uses two data structures, an *agenda* and a *chart*. The agenda is implemented as a priority queue with support for an *increase-key* operation. Concretely, we use a Fibonacci heap (Fredman and Tarjan, 1987), but many other types of heap could be used for the same purpose.

The chart is currently organised similarly to that of standard CKY in that it constitutes the upper-triangular portion of an (n + 1) x (n + 1) matrix, where n is the length of the string, and each cell $[i, j]$ in this matrix references some span from position i to position j in the input string. However, whereas in standard CKY, these cell indices reference the span of an entire expression, in our MG parser they reference only an expression's *narrow yield*, i.e. all those indices which are not part of some span which is undergoing or may undergo movement. For example, the narrow yield of the TP expression in 4 below is the set of indices corresponding to the words *Jack* and *gone there* (shown in bold face). The moving chain *why* is excluded from the narrow yield, as is the head string *has* because, depending on the type of complementizer which selects for this TP, *has* may un-

dergo head movement to yield *why has Jack gone there*, or not undergo head movement to yield, e.g., *you know why Jack has gone there*.

(4) **Jack**, has, **gone there** : t, why : -wh

Expressions within each cell are also currently placed into bins according to the first feature of their head chain, so that when the system encounters a t= feature, for example, it only needs to consider merging this expressions with other expressions whose first feature is t.

The call $\mathrm{updateWeight}(agenda, item)$ finds the current (backpointer, weight) pair of *item* in the agenda and compares it to the newly constructed (backpointer, weight) pair. The weight includes both the inside and outside scores. Only the pair with a lower weight is kept in the agenda. This update is made efficient by using an additional hash-table and the *increase-key* heap operation.

B Appendix: MGbank analyses of relative clauses and tough movement

The MGbank analysis of restrictive relative clauses is illustrated in phrase structural terms for the phrase *the book of ghost stories which Jack read* in Figure 3; the derivation tree for the simpler phrase *the book which Jack read* is shown in Figure 4. This analysis is inspired by an analysis in Bhatt (2002) and departs from that of Kayne (1994), where the wh determiner and the NP form a constituent in both the deep and surface structure (with the NP moving to the specifier of the wh DP to derive the correct word ordering). One reason for preferring Bhatt's analysis is that the wh item appears to form a constituent with the rest of the clause, as evidenced by the fact that it can form a conjunct with it: *the book* [*which Jack wrote*] *and* [*which Mary read*]. Bhatt (pages 79-81) suggests that the head noun moves to the left periphery and projects an NP layer over the clause, but does not specify what features drive this movement. MGbank uses a null type changing [relativizer] head to introduce a -n licensee onto the head noun which is then attracted to the +N of a [nom] head that selects the clause as its complement and projects the clausal NP layer. The [nom] head is needed here because in the MGbank formalism it is only possible for a specifier to project its fine-grained selectional properties and requirements (MASC, +3SG, -INF etc), not its selectee (n) category, hence the type of *projecting movement* Bhatt proposes must be precompiled into the lexicon.

Note that relative *that* is often treated as a complementizer rather than as a relative pronoun in MP (Radford, 2004, pages 228-230). When present in MGbank relatives, it therefore appears in the slot occupied by the null [decl] head in Figure 3, with a null [wh] head playing a similar role to the overt wh item in this example (selectional restrictions ensure that the grammar does not overgenerate examples like *the book which that Jack read* which violate the Doubly Filled Comp Filter (Chomsky and Lasnik, 1977)). Free relatives, as in *I like* [*what you're reading*], have a very similar analysis, but project only as far as CP (as they lack any head noun) and are then selected for by a null determiner head. Appositive relatives, as in *the book, which you've read, is on the table*, receive a head external analysis, again projecting only as far as CP and then adjoining to their head noun.

Figures 5 and 6 show the phrase structure and derivation trees for the tough movement example *that got hard to take*, which is one of the two examples of tough movement found in section 00 of the PTB, where it is embedded inside the larger sentence *"that got hard to take," he added*. It has generally been assumed since (Chomsky, 1977, 1981) that the infinitival clause is a type of relative clause with a null constituent in its left periphery that is co-indexed both with the object trace and the subject of the tough adjective. This null constituent is in fact included in the original PTB, although it is generally just ignored by treebank parsers. MGbank follows Brody (1993) and Hornstein (2001) in treating it as a trace of movement.

C Appendix: The PTB section 00 relative clause examples

Figures 7 and 8 show all the examples of free and non-free (non-reduced) relative clauses in section 00 of the PTB, and indicate which ones our best models did and did not correctly analyse.

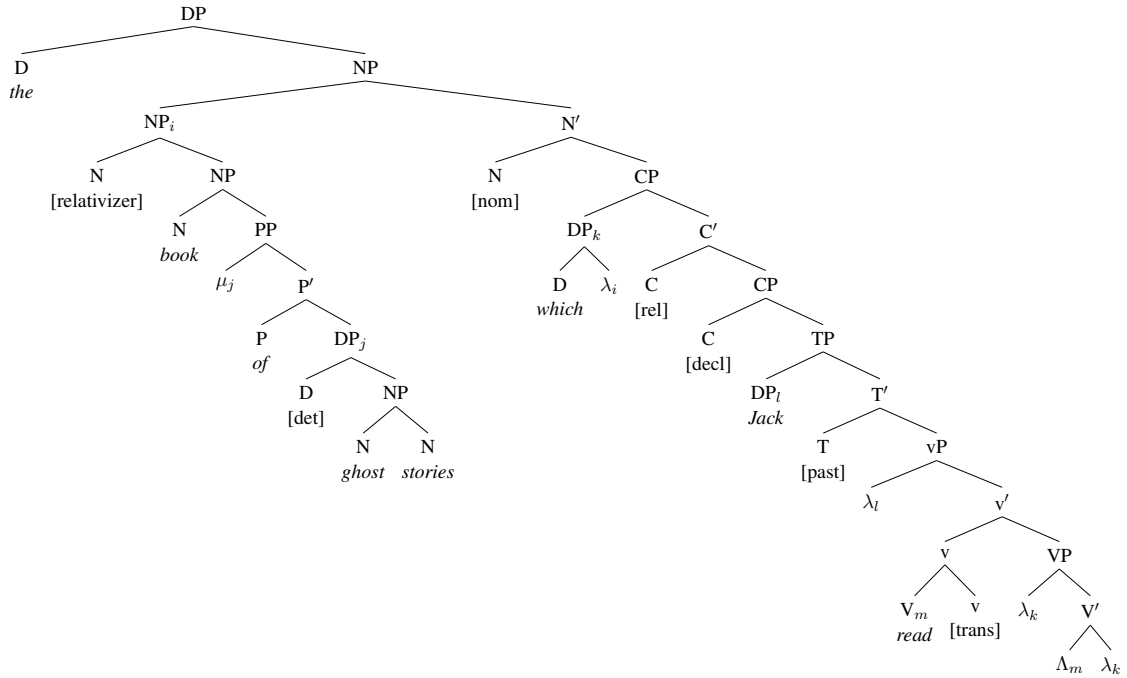

Figure 3: MGbank's phrase structural analysis of the phrase *the book of ghost stories which Jack read*, which contains a restrictive relative clause as the complement of the determiner *the*. The tree has been simplified in certain respects, for instance by removing the successive cyclic wh movement through spec-vP which is assumed in MP and included in the actual MGbank trees. Λ indicates a trace of head movement, λ indicates a trace of overt phrasal movement, and μ indicates the landing site of a covert movement.

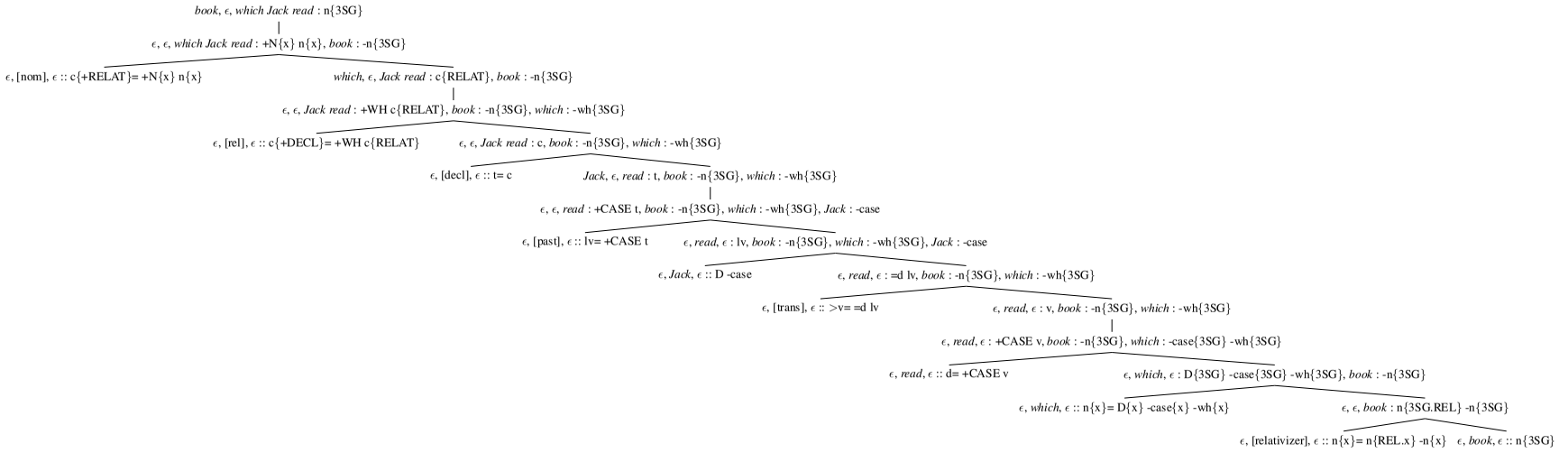

Figure 4: A derivation tree for the bracketed NP in *the* [*book which Jack read*]. Irrelevant selectional and agreement features are omitted to save space.

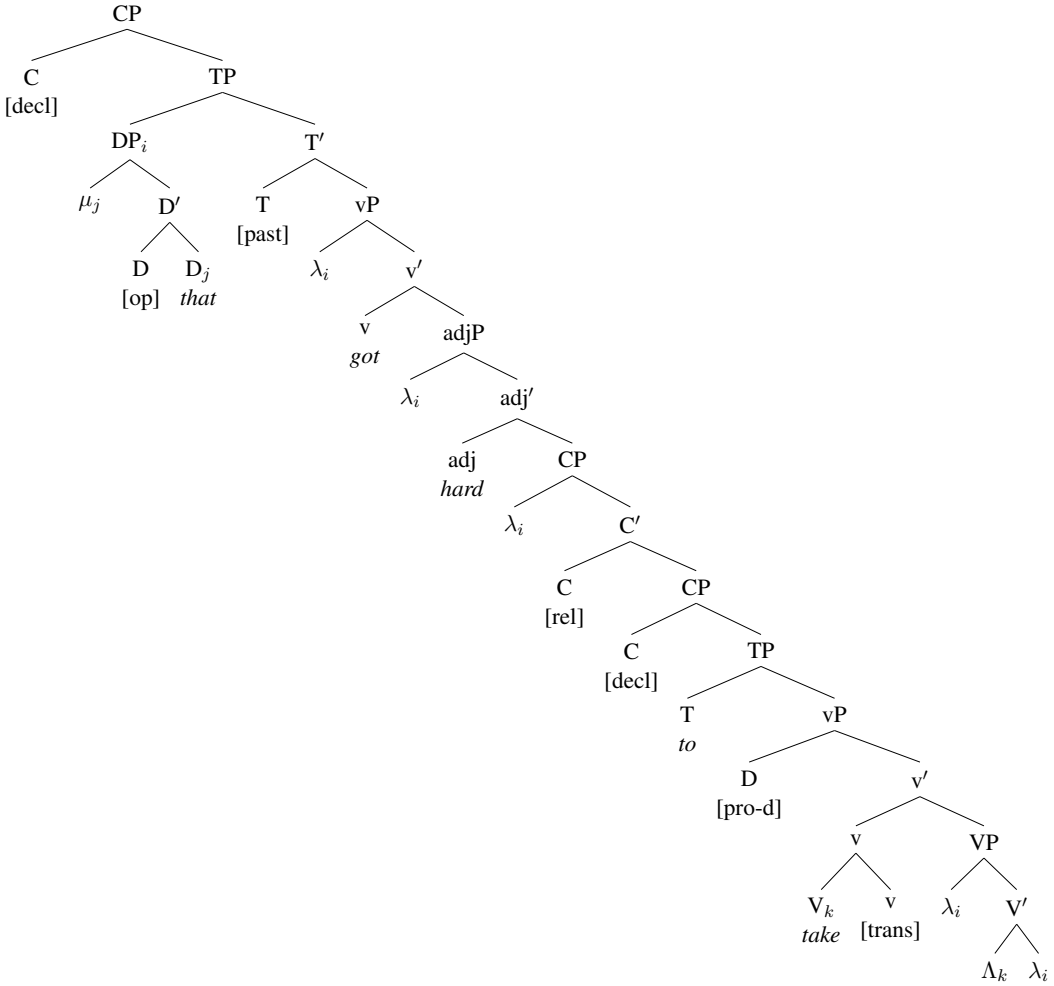

Figure 5: Derived Xbar tree showing MGbank's analysis for the phrase *that got hard to take* with tough movement. The tree has been simplified here by removing the successive cyclic wh movement through spec-vP that is standardly assumed in MP and is included in the actual MGbank trees. Note that μ indicates the landing site of a covert movement.

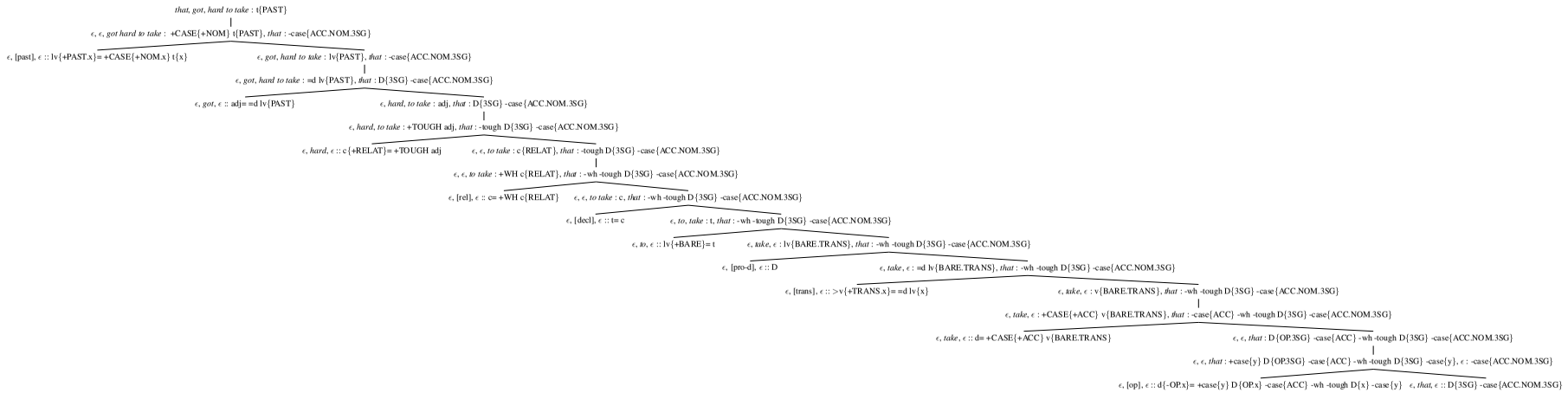

Figure 6: MG derivation tree for the sentence *that got hard to take*. Lowercase licensors, such as +case, trigger covert movement (see Torr and Stabler 2016).

✓	1.	The survey found that nearly half of Hong Kong consumers espouse *what* it *identified* as materialistic values compared with about one-third in Japan and the U.S.
✓	2.	*What* she *did* was like taking the law into your own hands
✓✓	3.	We work damn hard at *what* we *do* for damn little pay and *what* she *did* cast unfair aspersions on all of us
✓	4.	There may be others doing *what* she *did*
✓	5.	The U.S. wants the removal of *what* it *perceives* as barriers to investment ; Japan denies there are real barriers
✓	6.	But they have n't clarified *what* those might *be*
✓	7.	Deregulation has effectively removed all restrictions on *what* banks can *pay* for deposits as well as opened up the field for new products such as high - rate CDs
✓	8.	Mr. Martin said they have n't yet decided *what* their next move would *be* but he did n't rule out the possibility of a consent solicitation aimed at replacing Georgia Gulf 's board
✓	9.	What matters is *what* advertisers are *paying* per page and in that department we are doing fine this fall said Mr. Spoon
w.o.	10.	*What* this *tells* us is that U.S. trade law is working he said
t.o.	11.	The paper accused him of being a leading proponent of peaceful evolution a catch phrase to describe *what* China believes *is* the policy of Western countries to seduce socialist nations into the capitalist sphere
t.o.	12.	Despite the harsh exchanges the U.S. and China still seem to be looking for a way to mend relations which have deteriorated into *what* Mr. Nixon *referred to* as the greatest crisis in Chinese - American relations since his initial visit to China num years ago
✓	13.	Judge Ramirez num said it is unjust for judges to make *what* they *do*
✓	14.	Judges are not getting *what* they *deserve*
t.o.	15.	Composer Marc Marder a college friend of Mr. Lane 's who earns his living playing the double bass in classical music ensembles has prepared an exciting eclectic score that tells you *what* the characters are *thinking* and *feeling* far more precisely than intertitles or even words would
✓	16.	We have and I 'm sure others have considered *what* our options *are* and we 've had conversations with people who in the future might prove to be interesting partners

Figure 7: The 16 sentences with free object relative clause dependencies in section 00 of the PTB. Each tick indicates a point awarded for the correct identification of the extraction site of the wh word; t.o. indicates that the parser timed out before returning a parse, and w.o. indicates that the parser correctly identified an object relative dependency but extracted the wrong object of a double object verb. Our Abstract parser correctly identified 13/17 dependencies with a precision of 13/14. Our A* CCG parser correctly recovered 15.5/17 of these dependencies with precision 15.5/17 (we awarded the CCG parser half a point for sentence 15 because it related *what* to *thinking* but not *feeling*, which it analysed as intransitive). Note that sentence 3 contains two free object relative clauses.

	1.	It 's the petulant complaint of an impudent *American whom* Sony *hosted* for a year while he was on a Luce Fellowship in Tokyo – to the regret of both parties
✓	2.	It said the *man whom* it did not *name* had been found to have the disease after hospital tests
✓	3.	Commonwealth Edison now faces an additional court-ordered *refund* on its summerwinter rate differential collections that the Illinois Appellate Court has *estimated* at $ num million
✓	4.	But Rep. Marge Roukema -LRB- R. N.J -RRB- instead praised the House 's acceptance of a new youth training wage a *subminimum* that GOP administrations have *sought* for many years
	5.	Democratic Lt. Gov. Douglas Wilder opened his gubernatorial battle with Republican Marshall Coleman with an abortion *commercial* produced by Frank Greer that analysts of every political persuasion agree *was* a tour de force
✓	6.	Against a shot of Monticello superimposed on an American flag an announcer talks about the strong *tradition* of freedom and individual liberty that Virginians have *nurtured* for generations
	7.	Another was Nancy Yeargin who came to Greenville in num full of the *energy* and *ambitions* that reformers wanted to *reward*
	8.	Mostly she says she wanted to prevent the *damage* to self - esteem that her low - ability students would *suffer* from doing badly on the test
✓	9.	Mrs. Ward says that when the cheating was discovered she wanted to avoid the morale - damaging public *disclosure* that a trial would *bring*
✓	10.	Mr. Sherwood speculated that the *leeway* that Sea Containers *has* means that Temple would have to substantially increase their bid if they 're going to top us
	11.	A high - balance *customer* that banks *pine for* she did n't give much thought to the rates she was receiving nor to the fees she was paying
✓	12.	Interviews with analysts and business people in the U.S. suggest that Japanese capital may produce the economic *cooperation* that Southeast Asian politicians have *pursued* in fits and starts for decades
	13.	Interpublic Group said its television programming *operations – which* it *expanded* earlier this year – agreed to supply more than num hours of original programming across Europe in num
✓	14.	Interpublic is providing the programming in return for advertising *time which* it said *will* be valued at more than $ num million in num and $ num million in num
	15.	Mrs. Hills said many of the num *countries* that she *placed* under varying degrees of scrutiny have made genuine progress on this touchy issue
✓	16.	The Japanese companies bankroll many small U.S. companies with promising products or ideas frequently putting their money behind *projects* that commercial banks wo n't *touch*
	17.	In investing on the basis of future transactions a role often performed by merchant banks trading companies can cut through the *logjam* that small - company owners often *face* with their local commercial banks
✓✓	18.	He described the situation as an escrow *problem* a timing *issue which* he said *was* rapidly rectified with no losses to customers
	19.	In CAT sections where students ' knowledge of two - letter consonant sounds is tested the authors noted that Scoring High concentrated on the same *sounds* that the test *does* – to the exclusion of other *sounds* that fifth graders should *know*
✓✓	20.	The events of April through June damaged the *respect* and *confidence* which most Americans previously *had* for the leaders of China

Figure 8: The 20 sentences with non-free object relative clause dependencies in section 00 of the PTB. Our reified parser correctly recovered 13/24 of these (with precision of 13/17) by using a tag dictionary threshold initially set to 5. If the parser did not find a parse, then this was increased to 10 and the sentence reparsed. If a parse was still not found, the tag dictionary was turned off completely and a final parse attempted (on the single run, with no tag dictionary, our abstract parser performed best, retrieving 10/24 dependencies; the CCG A* parser returned 15/24 with precision 15/20).