Lecture 8

A Move-Eager Parser for MGs

The context-free parser for MGs has two disadvantages. First, it freely infers feature tuples, with the only restriction being that the features of these tuples are features of the grammar. Consequently, the parser may conjecture tuples that never occur in a well-formed derivation of the grammar, which is a major waste of resources. Second, its search through the tree is not contingent on conjectured landing sites. The parser follows a simple pattern of "specifiers before complements" pattern, with the minor twist that if a mover is found, the parser inserts it at the final target site and continues parsing from there. This means that the parser has to store a lot of material in memory. If it progressed on the most direct path to where it expects the mover to originate, without building any of the structure along the other paths, it may safe quite some resources. Let's call the first type of parser *Merge-eager*, and the second *Move-eager*. The top-down MG parser defined in Stabler (2012) is Move-eager and operates directly on the lexicon, which prevents it from conjecturing useless feature tuples.

8.1 A Closer Look at the Stabler Parser

See the paper for details, in particular the appendix. The crucial insights are as follows:

- Every Minimalist lexicon can be represented as a *prefix tree*, where the prefixes correspond to the feature components of lexical items read from right to left.
- The prefix tree guides the parser in the structure building process.
- In order to deal with non-determinism, the parser keeps multiple parses in memory.
- Parses are ordered by their probability, and those with a probability below a fixed threshold p are discarded. This is called *beam parsing*.
- Since the probability of a tree decreases with every level of embedding, left branch recursion is no longer a problem for the parser. Instead of adding more and more levels of embedding *ad infinitum*, the parser eventually reaches the probability threshold p and conjectures no further levels.

8.2 Psycholinguistic Adequacy

8.2.1 Generalizations about Relative Clauses

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

(2) a. The reporter$_i$ $[_{RC}$ who t_i attacked the senator] admitted the error.
  b. The reporter$_i$ $[_{RC}$ who the senator attacked $t_i]$ admitted the error.

We can use these sentences as test cases to determine whether the two types of MG parsers — Merge-eager and Move-eager — can account for these discrepancies, and if so, which one of them.

8.2.2 Refining our Metrics

In previous evaluations of parser predictions we used three metrics: payload for the number of nodes kept in memory, MaxTen for the maximum number of steps a node is kept in memory, and SumTen as the sum of all non-trivial tenure. For the MG-parser, it makes sense to introduce further subtypes of these. Since MGs have empty heads, the linguistic status of which is contentious, each metric can have two values, the standard one computed over all nodes, and a more restricted one that ignores all nodes that are empty heads. For the sake of succinctness we write these values in the slashed format m/n, where m is the standard value and n the restricted one. Another distinction that might be useful is the one between lexical items and phrasal nodes, which is why each metric also has a subtype that only considers leafs in the derivation. These subtypes are indicated by a subscripted *Lex*. Overall, then, we have four metrics and each metric has a slashed value.

Payload number of all/pronounced nodes kept in memory for at least 3 steps (this is increased from our previous default of 2, following Graf and Marcinek 2014)

Payload$_{Lex}$ number of all/pronounced leaves kept in memory for at least 3 steps

Max greatest number of steps that any/some pronounced node is kept in memory

Max$_{Lex}$ greatest number of steps that any/some pronounced leaf is kept in memory

These metrics are taken from Graf and Marcinek (2014), where they are called **Box**,**BoxLex**, **Max**, and **MaxLex**, respectively.

8.2.3 Sentential Complements and Relative Clauses

The Merge-eager and Move-eager derivation trees for (1a) and (1b) are given in Fig. 8.1–8.4. For the sake of clarity lexical items are given without their features, movement is indicated by dashed branches, and interior nodes are labeled in the fashion of X′-theory.

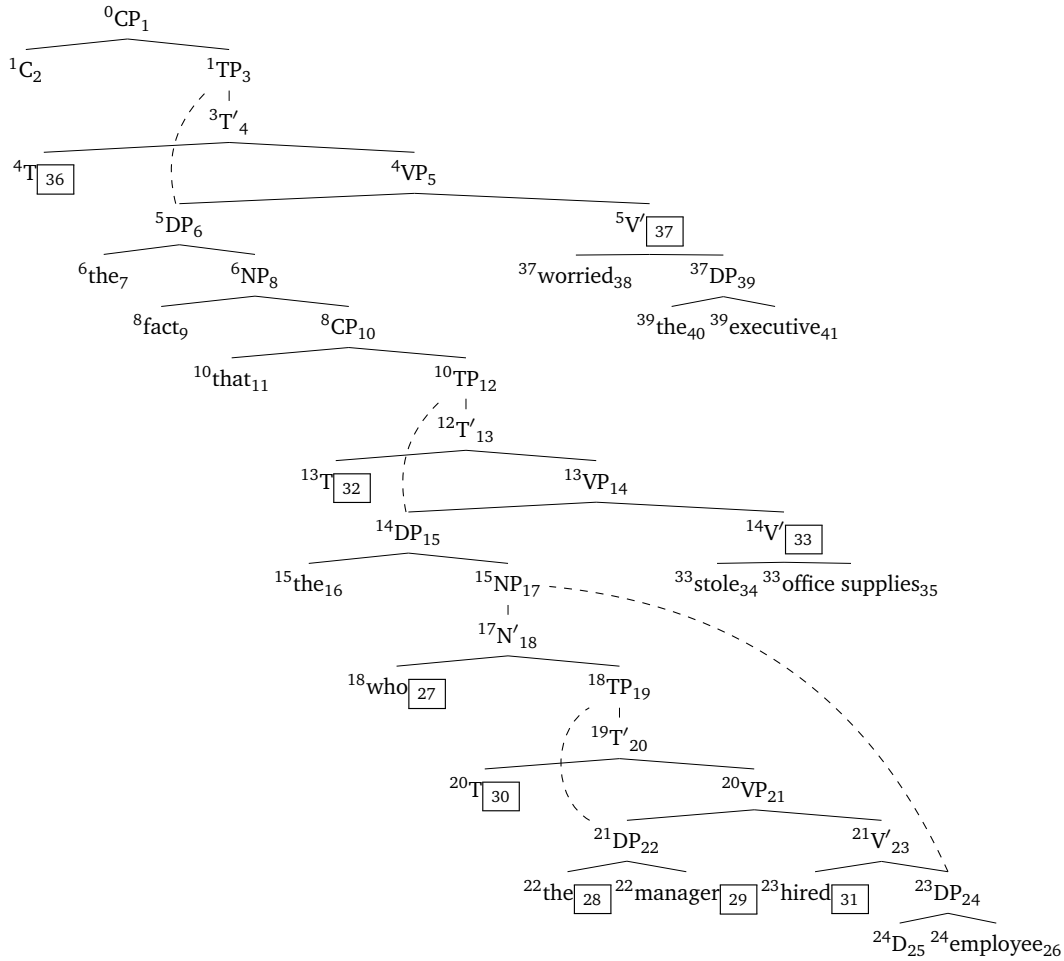

Figure 8.1: Merge-eager parse of sentential complement with embedded relative clause; **Max** = 32/32, $\textbf{Max}_{\textbf{Lex}}$ = 32/9, **Payload** = 9/6, $\textbf{Payload}_{\textbf{Lex}}$ = 7/4

As can be seen in Tab. 8.1, the Merge-eager and the Move-eager parsers behave almost exactly the same. With both of them **Max** makes barely a difference between the two structures, whereas $\textbf{Max}_{\textbf{Lex}}$ correctly prefers SC/RC if empty heads are ignored. Results are mixed for the payload metrics. With a Merge-eager parser they indeed capture the preference for SC/RC, but the Move-eager parser shows no difference for these metrics. This isn't too surprising considering that payload wasn't a useful metric for the CFG parsing models either. The surprising thing, then, is that the payload metrics actually work for Merge-eager parser.

8.2.4 Subject Gaps and Object Gaps

The results for subject gaps versus object gaps are slightly different. Once again $\textbf{Max}_{\textbf{Lex}}$ makes the right predictions for both parsers, and so do **Payload** and $\textbf{Payload}_{\textbf{Lex}}$

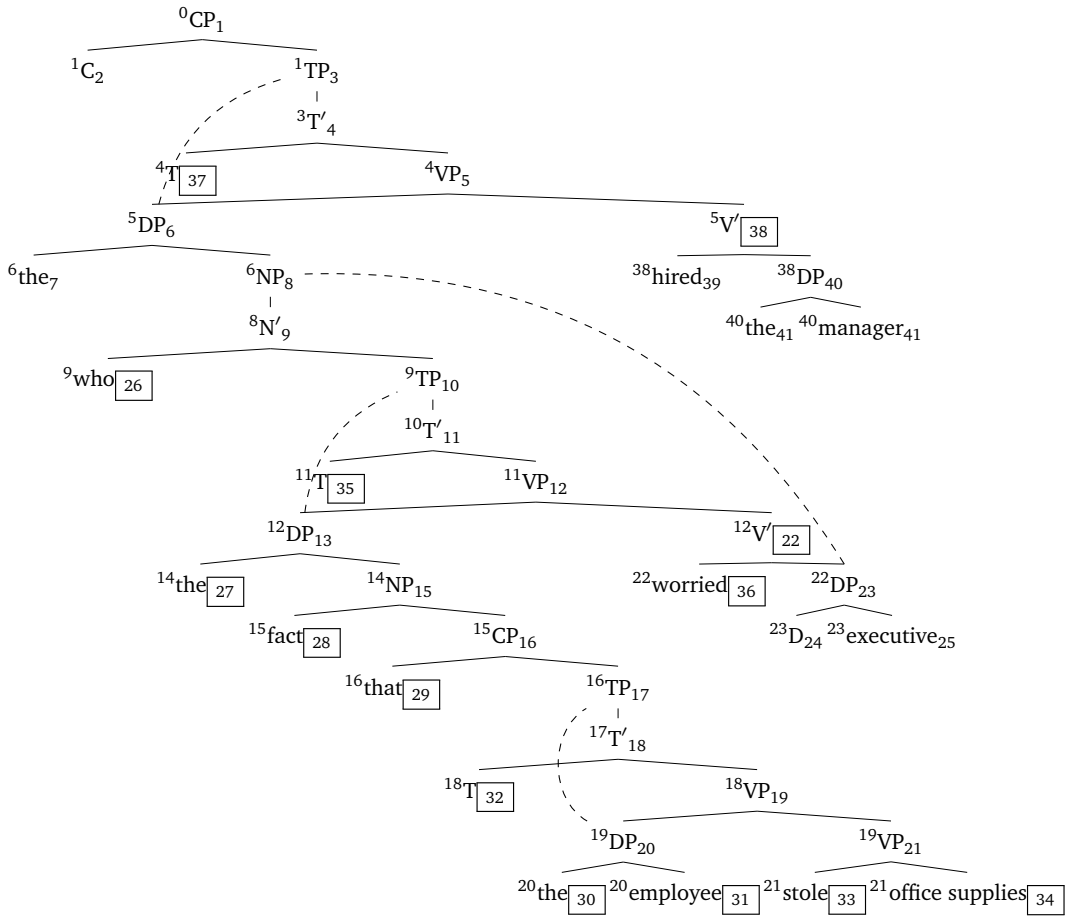

Figure 8.2: Merge-eager parse of relative clause containing a sentential complement; **Max** = 33/33, $\textbf{Max}_{\textbf{Lex}}$ = 33/17, **Payload** = 14/11, $\textbf{Payload}_{\textbf{Lex}}$ = 12/9

	SC/RC	**RC/SC**
Payload	9/6	14/11
$\textbf{Payload}_{\textbf{Lex}}$	7/4	12/9
Max	32/32	33/33
$\textbf{Max}_{\textbf{Lex}}$	32/9	33/17

(a) Merge-eager

	SC/RC	**RC/SC**
Payload	8/5	8/5
$\textbf{Payload}_{\textbf{Lex}}$	5/2	5/2
Max	31/31	32/32
$\textbf{Max}_{\textbf{Lex}}$	31/8	33/22

(b) Move-eager

Table 8.1: Overview of processing predictions for SC/RC and RC/SC

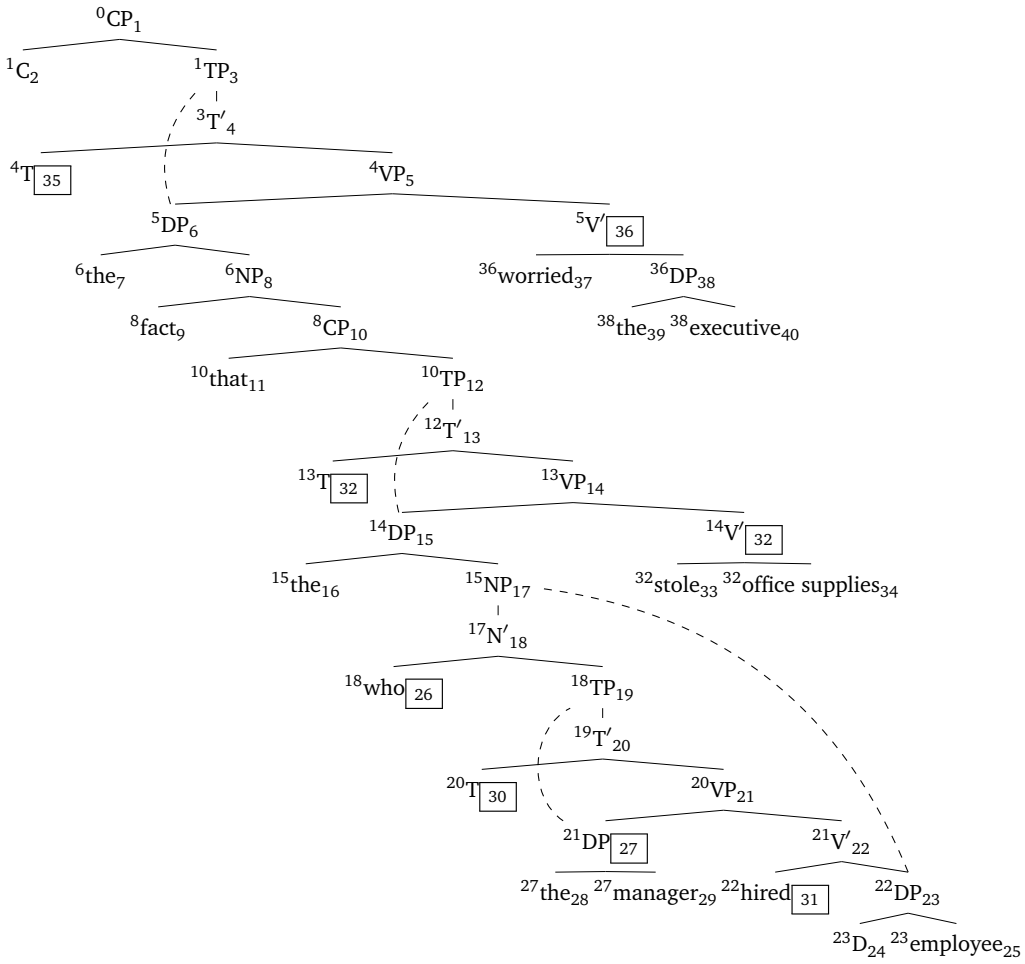

Figure 8.3: Move-eager parse of sentential complement with embedded relative clause; **Max** $= 31/31$, $\textbf{Max}_{\textbf{Lex}} = 31/8$, **Payload** $= 8/5$, $\textbf{Payload}_{\textbf{Lex}} = 5/2$

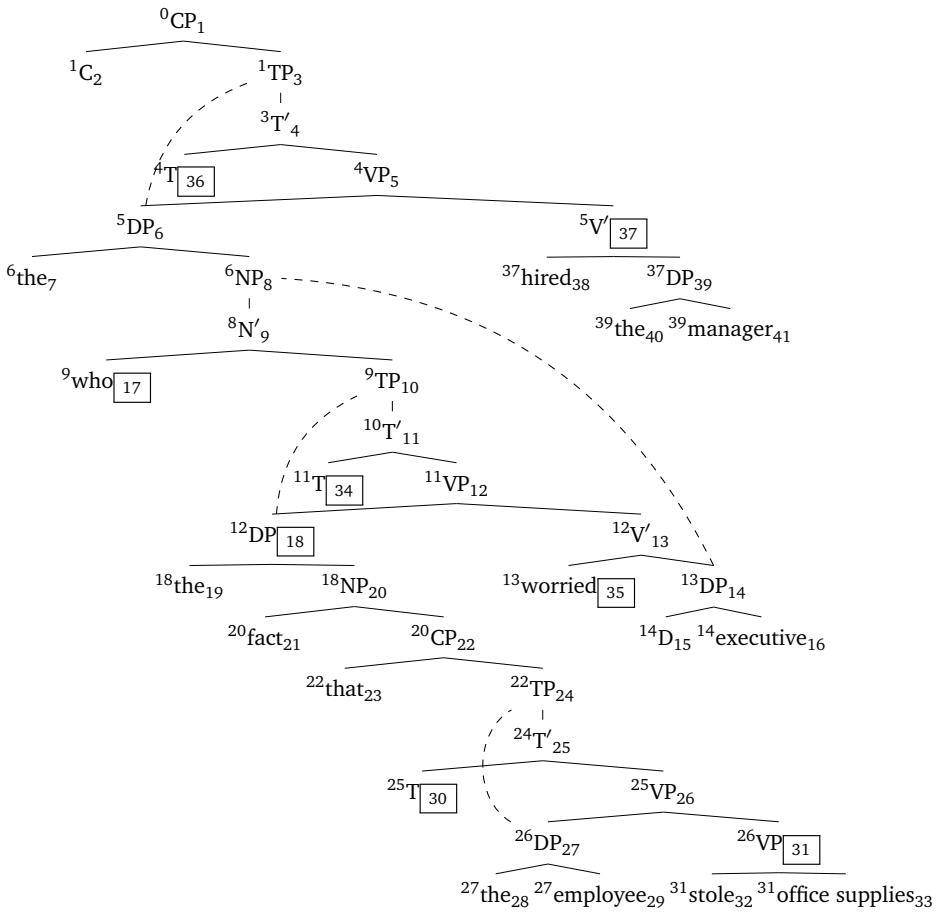

Figure 8.4: Move-eager parse of relative clause containing a sentential complement; **Max** = 32/32, $\textbf{Max}_{\textbf{Lex}}$ = 32/22, **Payload** = 8/5, $\textbf{Payload}_{\textbf{Lex}}$ = 5/2

(although the difference is much more pronounced with the Merge-eager parser in this case). **Max** seems to predict a tie, but earlier in the course we already said that if two trees have the same **Max** value, we can rank them according to the second-highest tenure value. With both parsers these values are 7/7 for the subject gap sentence and 10/9 for the object gap sentence, so a preference for subject gaps emerges even with **Max**.

	SubjRC	**ObjRC**
Payload	5/3	7/5
Payload$_{\text{Lex}}$	3/1	6/4
Max	19/19	19/19
Max$_{\text{Lex}}$	19/7	19/9

(a) Merge-eager

	SubjRC	**ObjRC**
Payload	5/3	6/4
Payload$_{\text{Lex}}$	3/1	4/2
Max	19/19	19/19
Max$_{\text{Lex}}$	19/7	19/9

(b) Move-eager

Table 8.2: Overview of processing predictions for SC/RC and RC/SC

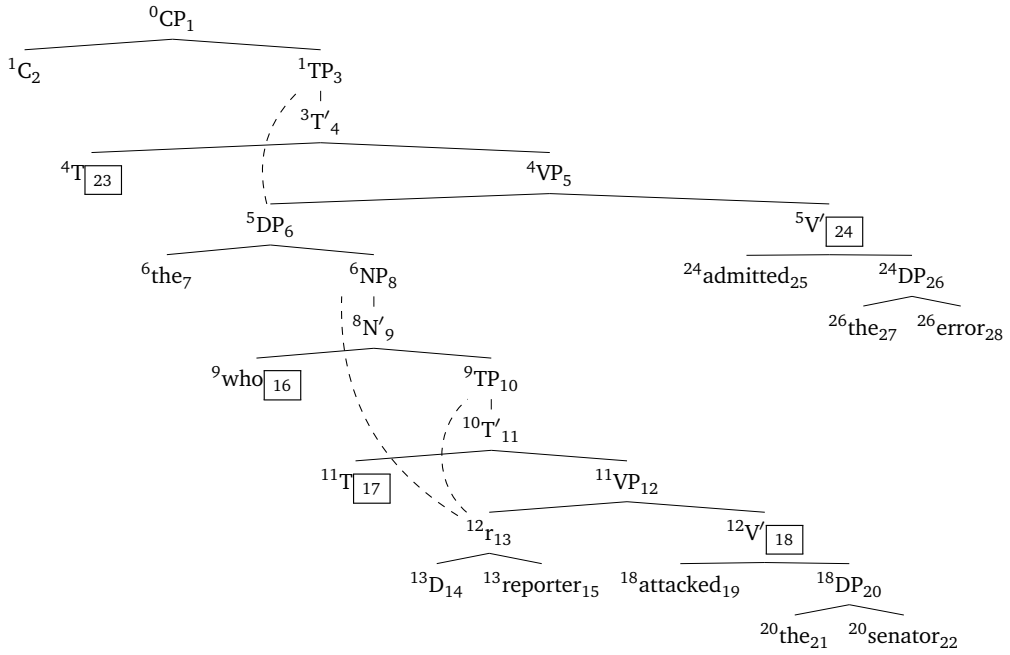

Figure 8.5: Merge-eager and Move-eager parse of relative clause with subject gap; **Max** = 19/19, **Max$_{\text{Lex}}$** = 19/7, **Payload** = 5/3, **Payload$_{\text{Lex}}$** = 3/1

References and Further Reading

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

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

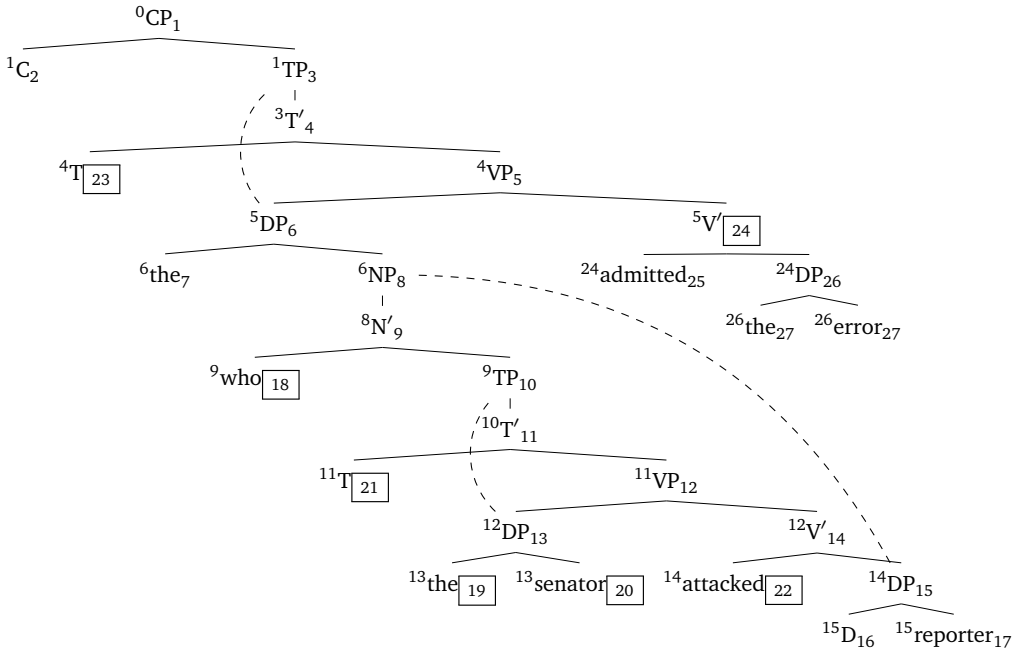

Figure 8.6: Merge-eager parse of relative clause with object gap; **Max** = 19/19, $\textbf{Max}_{\textbf{Lex}}$ = 19/9, **Payload** = 7/5, $\textbf{Payload}_{\textbf{Lex}}$ = 6/4

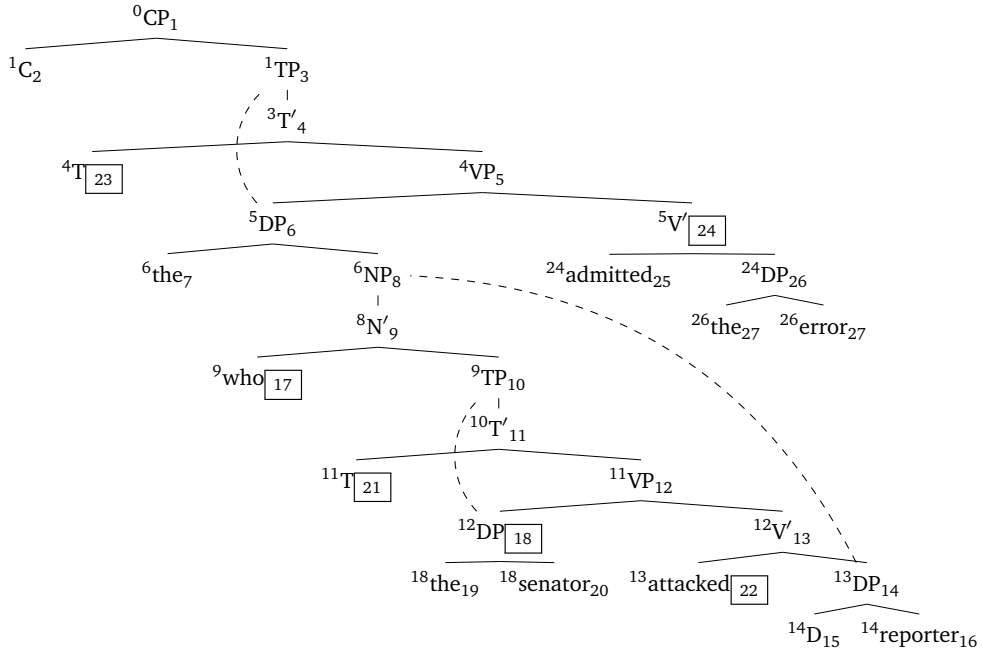

Figure 8.7: Move-eager parse of relative clause with object gap; **Max** = 19/19, $\textbf{Max}_{\textbf{Lex}}$ = 19/9, **Payload** = 6/4, $\textbf{Payload}_{\textbf{Lex}}$ = 4/2

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