Proceedings of the Society for Computation in Linguistics

Volume 2 Article 24

2019

Evaluation Order Effects in Dynamic Continuized CCG: From Negative Polarity Items to Balanced Punctuation

Michael White
The Ohio State University, mwhite@ling.osu.edu

Evaluation Order Effects in Dynamic Continuized CCG: From Negative Polarity Items to Balanced Punctuation

Michael White
Department of Linguistics
The Ohio State University
Columbus, OH 43210 USA
mwhite@ling.osu.edu

Abstract

Combinatory Categorial Grammar's (CCG; Steedman, 2000) flexible treatment of word order and constituency enable it to employ a compact lexicon, an important factor in its successful application to a range of NLP problems. However, its word order flexibility can be problematic for linguistic phenomena where linear order plays a key role. In this paper, we show that the enhanced control over evaluation order afforded by Continuized CCG (Barker & Shan, 2014) makes it possible to not only implement an improved analysis of negative polarity items in Dynamic Continuized CCG (White et al., 2017) but also to develop an accurate treatment of balanced punctuation.

1 Introduction

Combinatory Categorial Grammar (CCG; Steedman, 2000) has been increasingly employed with success for a wide range of NLP problems. An important factor in its success is that its flexible treatment of word order and constituency enable it to employ a compact lexicon, making it easier to acquire lexicalized grammars automatically (Artzi and Zettlemoyer, 2013) and to train machine-learned models that generalize well (Clark and Curran, 2007; Lee et al., 2016). However, its word order flexibility can be problematic for linguistic phenomena where linear order plays a key role. In particular, linear order effects can be problematic for Steedman's (2012) treatment of negative polarity items (NPIs) as well as for implementing a treatment of balanced punctuation that attempts to track the status of the right periphery, as White & Rajkumar (2008) have previously shown.

In this paper, we show that the enhanced control over evaluation order afforded by Continuized CCG (Barker & Shan, 2014) makes it possible to successfully address both of these problems. In particular, we show that Barker & Shan's analysis of NPIs, which does not suffer from the linear order issues that we raise for Steedman's analysis, can be straightforwardly implemented in Dynamic Continuized CCG (White et al., 2017), a system that combines Barker & Shan's "tower" grammars with Charlow's (2014) monadic approach to dynamic semantics, employing Steedman's CCG on the tower bottom for predicate-argument structure. We then go on to show that the same technique used in Barker & Shan's analysis of NPIs can be used to successfully handle balanced punctuation, taking advantage of Charlow's monadic semantics to implement a novel approach to NP appositives.

Barker & Shan's continuized grammars make crucial use of continuations, a technique developed in programming language semantics to control and analyze evaluation order, and which enables the analyses presented here. Charlow's dynamic semantics implements an alternative to Steedman's (2012) approach to the exceptional scope of indefinites that fits naturally with ongoing work in dynamic semantics. Meanwhile, by combining the Barker-Shan-Charlow approach with Steedman's CCG for predicate-argument structure, the resulting system respects Steedman's Principle of Adjacency, making it attractive for computational implementations. As such, we suggest that the framework offers a promising starting point for work that tackles complex linguistic phenomena in a practical way. A prototype implementation verifying the analyses accompanies the paper.[1]

2 Negative Polarity Items in CCG

Steedman (2012) develops an approach to quantification in CCG, including a treatment of negative polarity items. Steedman's analysis follows

[1] https://github.com/mwhite14850/dyc3g

Proceedings of the Society for Computation in Linguistics (SCiL) 2019, pages 226-235.
New York City, New York, January 3-6, 2019

earlier treatments in categorial grammar (Dowty, 1994; Bernardi, 2002) in requiring negative polarity items to be licensed by an outscoping downward-entailing quantifier, negation or other licensor (e.g. verbs such as *deny*, which lexically incorporate negation). For example, Figure 1a shows how CCG can account for the difference between *No one enjoyed anything*, where the NPI *anything* is outscoped by its licensor *no one*, and **Kim enjoyed anything*, where the NPI *anything* has no licensor.[2] In Steedman's analysis, S° indicates neutral polarity, while $S^{\bullet}$ indicates reverse polarity. The NPI *anything* combines with the verb *enjoyed* on its left to yield $S^{\bullet}\backslash NP$, the category for a verb phrase with reverse polarity.[3] The downward-entailing quantifier *no one* here combines with an inverted-polarity VP on the right to yield S°, the category for a clause which again has neutral polarity. Root sentences are required to have positive polarity, which is compatible with neutral polarity. With **Kim enjoyed anything*, by contrast, the subject *Kim* does not license NPIs, and thus the derivation ends with $S^{\bullet}$, which is incompatible with root-level positive polarity.

As Barker & Shan observe, previous accounts in the categorial grammar literature have failed to account for the constraint that licensors must not only outscope NPIs but also precede them. Consequently, as illustrated with Steedman's account in Figure 1b, where *nothing* outscopes *anyone* but does not precede it, these accounts mistakenly predict that sentences such as **Kim gave anyone nothing* should be acceptable.[4]

3 NPIs in Continuized CCG

A continuized grammar is one where the meaning of expressions can be defined as a function on a portion of its surrounding context, or *continuation* (Barker, 2002; Shan and Barker, 2006; Barker and Shan, 2014). To make it easier to reason about continuized grammars, Barker & Shan devised the "tower" notation illustrated in Figure 2.[5] For example, *no one* has a tower category with NP on the bottom and two Ss on top; reading counterclockwise from the bottom, this category represents a constituent that acts locally as an NP, takes scope over an S^{-}, and returns an S°. Here S^{-} represents the subtype of negative clauses, while S° is again a neutral clause.[6] The semantics is $\lambda k.\neg\exists y.ky$, a function from a continuation k of type $e \rightarrow t$ to a quantified expression of type t. A continuized meaning of this form can be abbreviated by representing the location where the continuation argument applies with $[\,]$ and putting the argument to the continuation on the bottom of the tower, as shown.

In a continuized grammar, all expressions can potentially be given continuized meanings via the Lift ($\uparrow$) operation. This is illustrated in Figure 2a where the categories for *Kim* and *gave* are lifted, taking on the semantics $\lambda k.k(\mathrm{kim})$ and $\lambda k.k(\lambda yzx.\mathrm{give}(x,y,z))$, resp. Lifting the category for *gave* allows it to combine with that of *no one* using scopal combination. The way in which scopal combination works in the tower notation is shown in Figure 3b (left): on the tower top, the syntactic categories must match in the middle (shown as matching Es here), while semantically the continuized functions $g[\,]$ and $h[\,]$ compose in surface order, yielding $g[h[\,]]$; on the tower bottom, the categories A and B with semantics a and b combine as they normally would in CCG (using the combinators in Figure 3a), yielding a category C with semantics c.[7] In the example, $[]$ and $\neg\exists y.[]$ compose to again yield $\neg\exists y.[\,]$ on the tower top, with $\lambda yzx.\mathrm{give}(x,y,z)$ applying to y and yielding $\lambda zx.\mathrm{give}(x,y,z)$ on the bottom.

As Barker & Shan observe, the explicit lifting step seen in Figure 2a can be integrated with the scopal combination step, as shown in the other recursively defined rules in Figure 3b, thereby avoiding an infinite regress when applying the lifting rule. Figure 2b shows how Lift Left ($\uparrow\mathbf{L}$) can

[2] Here it's important to distinguish NPI *anything* from free-choice *anything*, where the latter is licensed instead in generic/habitual contexts, and thus can't appear in episodic sentences (as assumed in the example here).

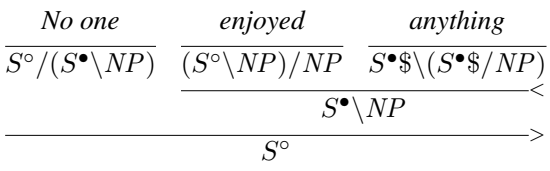
[3] In the category for *anything*, the dollar sign matches a stack of arguments, which in this case is the single subject argument.

[4] As Barker & Shan also observe, similar linear order effects arise with binding, which Steedman handles at the level of logical form; such an approach would not seem applicable here, given that NPI licensing is generally taken to have both semantic and syntactic components.

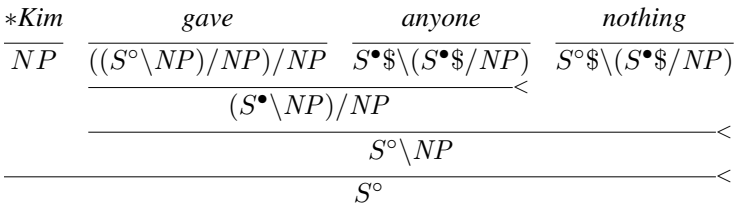
[5] Semantic types are generally suppressed in this and subsequent figures.

[6] The subtype of positive clauses, represented by S^{+}, is used with positive polarity items. Note that Barker & Shan's polarity notation differs from the Dowty-style notation Steedman adopts in order to reflect its status as grammaticized rather than purely semantic information.

[7] The combinator for combining two scopal terms m and n is $\lambda mnk.m(\lambda x.n(\lambda y.k(xy)))$, assuming forward application on the tower bottom. Formulating the rules recursively allows the base combinator to be factored out while also generalizing to multi-level towers.



(a)



(b)

Figure 1: Negative Polarity Item Licensing in CCG

be applied twice rather than using explicit lifting. The final representations are derived by collapsing the towers using the recursively defined Lower ($\downarrow$) operation in Figure 3c, which repeatedly applies the continuized semantics to the identity continuation $\lambda k.k$.

The polarity types in Figure 2 interact to correctly derive *Kim gave no one anything* and correctly rule out **Kim gave anyone nothing*. In Figure 2a, the licensor *no one* both precedes and outscopes the NPI *anything*, yielding a clause with neutral polarity, which is compatible with the root-level positive polarity requirement.[8] By contrast, in Figure 2b, the NPI *anyone* precedes its licensor, leading to a derivation that ends with S^-, which is incompatible with root-level positive polarity.[9] Figure 4 illustrates how the account makes the correct predictions even when Steedman's CCG, with its flexible approach to word order, is employed on the tower bottom: *Kim gave nothing to anyone* is correctly ruled in, and **Kim gave to anyone nothing* is correctly ruled out, even when the type-raised NPI PP (abbreviated as $PP^\uparrow$) swaps places places with its NP licensor using the backwards crossed composition combinatory rule ($< \mathbf{B}_\times$).

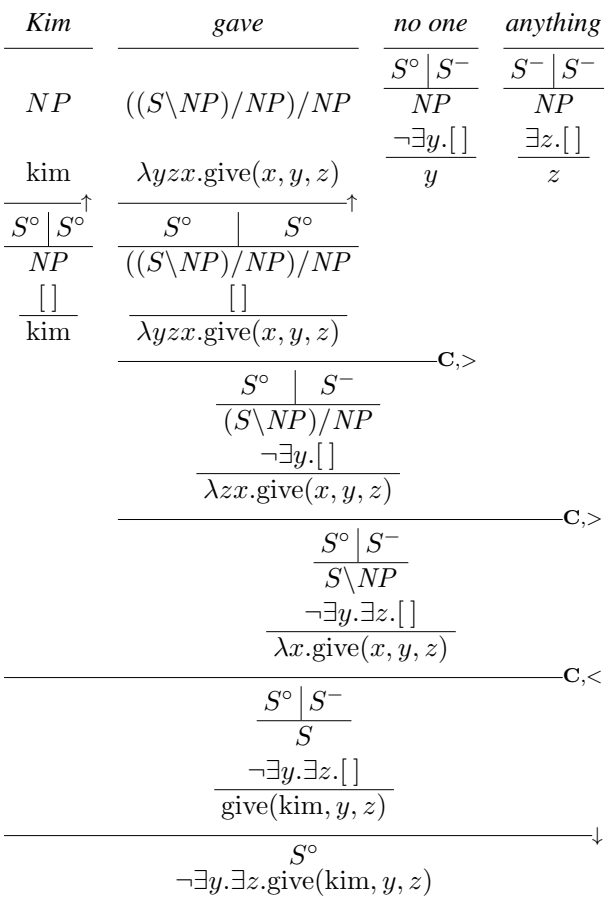
[8]The Lower rule is compatible with subtypes of S at the tower upper right.

[9]The derivation does not go through even if *nothing* inverts to outscope *anyone*, as the *nothing*'s negation-canceling effect ends up on the wrong level. See (Barker and Shan, 2014) for discussion.

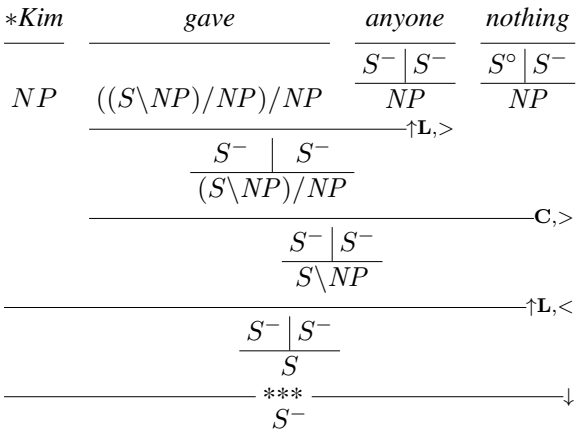
4 Flexible Word Order and Balanced Punctuation

To better serve the needs of generation, White and Rajkumar (2008) implement an approach to making the treatment of punctuation in a broad coverage grammar extracted from the CCGbank (2007) more precise by adding lexicalized punctuation categories to deal with constructions involving punctuation. In doing so, however, they observe that CCG's flexible treatment of word order is problematic when it comes to implementing a grammar-based approach to balanced punctuation, for reasons we review in this section. (Nunberg 1990 gives extensive arguments that writers have implicit knowledge of the grammar of punctuation that is comparable in intricacy to their grammatical knowledge of spoken language. Whether one agrees with Nunberg or not, there is practical value to grammars that treat punctuation with sufficient precision to satisfy the needs of parsing and generation.)

The original CCGbank corpus does not have lexical categories for punctuation; instead, punctuation marks carry categories derived from their part of speech tags and form part of a binary rule. It is assumed that there are no dependencies between words and punctuation marks and that the result of punctuation rules is the same as the non-punctuation category. Binary rules of this kind miss out on many linguistic generalizations, most glaringly the presence of manda-



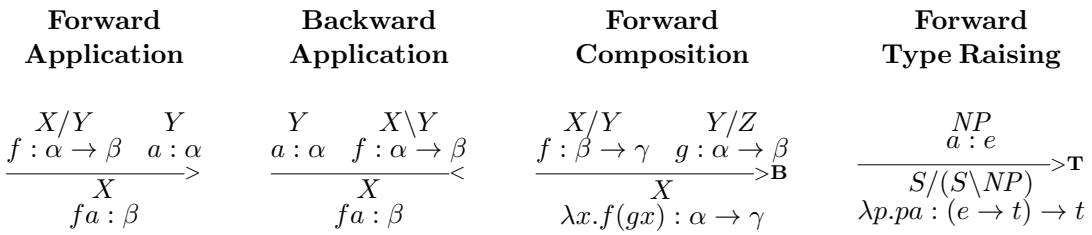
(a) Negative Polarity Item Licensing, with Explicit Lifting



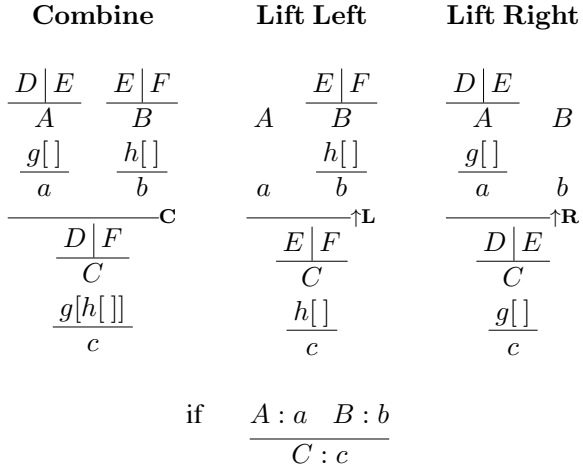
(b) Unlicensed Negative Polarity Item, with Integrated Lifting (Semantics Suppressed)

Figure 2: Continuized CCG Derivations

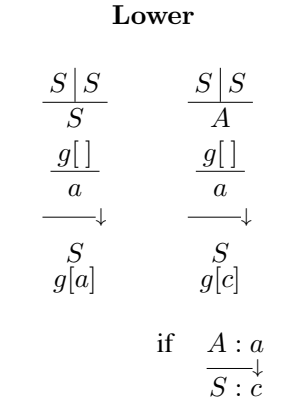
tory balancing marks in sentence-medial comma or dash adjuncts. For example, NP appositives can occur sentence medially or finally, and the conventions of writing mandate that sentence medial appositives should be balanced—i.e., the appositive NP should be surrounded by commas (or dashes) on both sides—while sentence final appositives should be unbalanced—i.e., they should only have one preceding comma or dash. The paradigm below illustrates:



(a) Base CCG Combinators (not exhaustive)



(b) Combination with Lifting



(c) Lowering (base and recursive)

Figure 3: Continuized CCG

(1) a. Kim, CEO of XYZ, loves Sandy.

b. *Kim, CEO of XYZ loves Sandy.

c. Kim loves Sandy, CEO of XYZ.

d. *Kim loves Sandy, CEO of XYZ,.

e. Kim loves Sandy, CEO of XYZ, madly.

f. *Kim loves Sandy, CEO of XYZ madly.

The literature discusses various means to address the issue of overgeneration: absorption rules (Nunberg, 1990), syntactic features (Doran, 1998) and (Briscoe, 1994) and semantic features (White, 2006). Nunberg (1990) argues that text adjuncts introduced by punctuation marks have an underlying representation where these adjuncts have marks on either side. They attain their surface form when a set of presentation rules are applied. This approach ensures that all sentence medial cases like (1a) and (1e) above are generated correctly, while unacceptable examples (1b) and (1f) would not be generated at all. Example (1c) would at first be generated as (1d): to deal with such sentences, where two points happen to coincide, Nunberg posits an implicit point which is absorbed by the adjacent point. Absorption occurs according to the "strength" of the two points. Strength is determined according to the Point Absorption Hierarchy, which ranks commas lower than dashes, semi-colons, colons and periods. As White (1995) observes, from a generation-only perspective, it makes sense to generate text adjuncts which are always balanced and post-process the output to delete lower ranked points, as absorption uses relatively simple rules that operate independently of the hierarchy of the constituents. However, using this approach for parsing would involve a preprocessing step which inserts commas into possible edges of possible constituents, as described in (Forst and Kaplan, 2006). To avoid this considerable complication, Briscoe (1994) has argued for developing declarative approaches involving syntactic features in bi-directional systems, with no deletions or insertions of punctuation marks.

Following Briscoe, White and Rajkumar implement the categories shown in Figure 5 for appos-

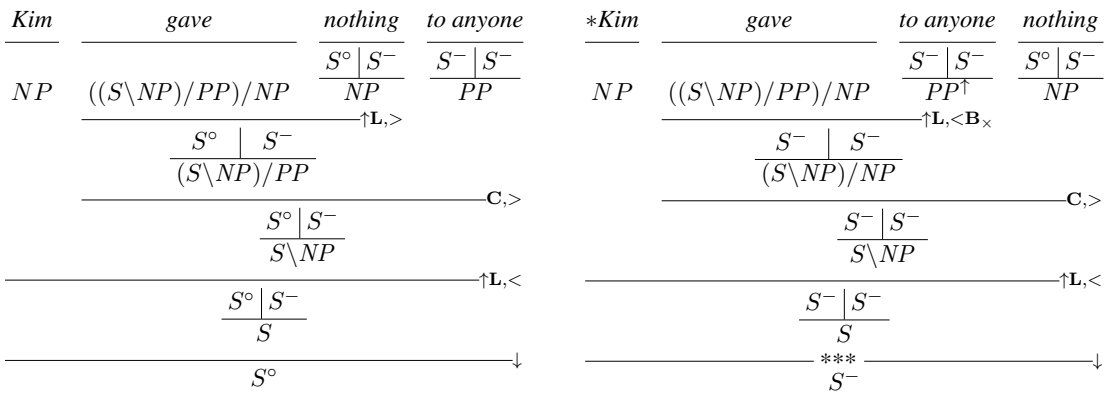


Figure 4: Word Order Flexibility and NPI Licensing

(2) a. , ⊢ $\mathsf{np}_{\langle 1\rangle bal=-,end=nil} \backslash \mathsf{np}_{\langle 1\rangle end=nil} /_{\star} \mathsf{np}_{\langle 3\rangle end=nil}$

b. , ⊢ $\mathsf{np}_{\langle 1\rangle bal=+,end=comma} \backslash \mathsf{np}_{\langle 1\rangle end=nil} /_{\star} \mathsf{punct}[,] /_{\star} \mathsf{np}_{\langle 3\rangle end=nil}$

Figure 5: Categories for Unbalanced and Balanced Appositive Commas in CCG (White and Rajkumar, 2008)

itive commas. Here, the unbalanced appositive comma has the category in (2a) where the comma selects as argument the appositive NP and converts it to a nominal modifier. For balanced appositives, the comma in (2b) selects the appositive NP and the balancing comma to form a nominal modifier. As these categories show, this approach involves the incorporation of syntactic features for punctuation (*bal* and *end*) into atomic categories so that certain combinations are blocked. To ensure proper appositive balancing sentence finally, the rightmost element in the sentence should transmit these features to the clause level, which the full stop can then check for the presence of right-edge punctuation; elsewhere, categories should ensure that their leftward arguments are balanced. The approach ensures that (1a)–(1f) above are all correctly generated or blocked.

The first issue with this aproach is that it does not work when crossing composition is used with adverbs in heavy-NP shift contructions, as illustrated in Figure 6.[10] Here the category for *loves* is intended to pass up the end punctuation feature from its direct object NP to the clause level (via the *PE* variable). Meanwhile, the category for *madly* is designed to combine with a balanced VP on the left to make a VP that has no end punctuation, which would be appropriate if *madly* appeared at the end of the verb phrase. However, when this category is used with backwards crossing composition as shown here, the result is category that claims to have no end punctuation when in fact it ends in a comma.

The second issue with the approach is that it is not adequate to deal with extraction involving ditransitive verbs, as shown in Figure 7. Here the comma at the end of the relative clause is not propagated to the root level. This is because the *end* feature for the relative clause should depend on the first (indirect) object of *gave*, rather than the second (direct) object as in a full ditransitive clause. Since CCG uses the same category for main and relative clauses, however, the right end punctuation is not correctly tracked.

As an interim solution to avoid overgeneration in such cases, White and Rajkumar (2008) implement an ad hoc post-filter on derivations to eliminate improperly balanced punctuation. In the next section, we'll see that continuized CCG makes it possible to successfully incorporate such constraints into the grammar itself.

[10] The appositive here is shortened to just *CEO* for illustration; naturally the object NP would need to be longer to count as a felicitous heavy NP.

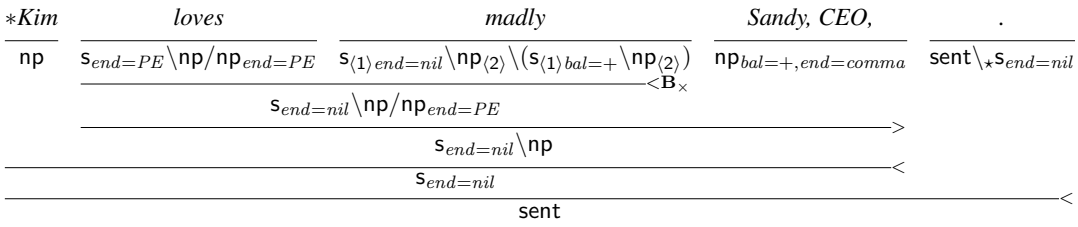


Figure 6: Crossing Composition and End Punctuation Tracking Issue in CCG

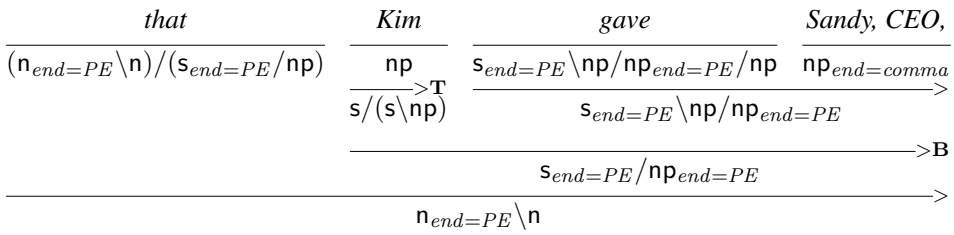


Figure 7: Object Extraction and End Punctuation Tracking Issue in CCG

5 An Evaluation Order Account of Balanced Punctuation

The approach presented in Section 3 to using the continuation layer to handle linear order constraints on NPI licensing can be generalized to also successfully track punctuation at the right periphery. As shown earlier, the category at the top left of the tower can be used to impose requirements on the preceding context, while the top right category can track information made available to the subsequent context. The idea for handling balanced punctuation is illustrated in Figure 8, where a bad comma-period sequence is correctly blocked despite the use of backwards crossed composition for heavy-NP shift. Tower categories, such as the one for *Sandy, CEO,*, ordinarily require their leftward context to have balanced punctuation, as shown with the top-left category S^{bal}; the phrase as a whole is balanced and ends with a comma, as the category S^{bal}_{comma} attests. Punctuation information on tower categories is propagated through the continuation layer via the Combination, Lift Left and Lift Right rules;[11] here, the direct object NP punctuation info is propagated up to the clausal level. At this point, combination with the full stop is blocked, since the full stop seeks a tower on the left that has no end punctuation, but the clause ends in a comma.[12] With relative clauses (not shown), there is no problem tracking the punctuation at the right edge since this information is passed along the tower top rather than via the arguments of the CCG categories on the tower bottom, as in the problematic Figure 7.

The comma category for deriving balanced appositives is given in (3b) in Figure 9; this is the one whose result is shown in Figure 8. The corresponding unbalanced comma category—the one that would lead to a grammatical derivation—appears in (3a).

Semantically, with either comma category, the semantics of the predicative NP is added to that of the modified NP, taking advantage of the capacity of the monadic dynamic semantics to sequence constraints on entities. To show how this works, we first briefly introduce the basic idea of Charlow's monadic semantics, following White et al's (2017) presentation.

Charlow's (2014) dynamic semantics makes use of the State.Set monad (Hutton and Meijer, 1996), which combines the State monad for handling side effects with the Set monad for non-determinism. The State monad pairs ordinary semantic values with a state, which is threaded through computations. The Set monad models non-deterministic choices as sets, facilitating a non-deterministic treatment of indefinites. For example, the dynamic meaning of *a linguist swims* appears in (4): here, the proposition that x swims, where x is some linguist, is paired with a state that augments the input

[11] The Lift and Lift Left rules are modified to require balanced punctuation on the left.

[12] For readability, Lambek-style (result-on-top) slashes are used at the tower level, rather than Steedman-style (result-first) ones.

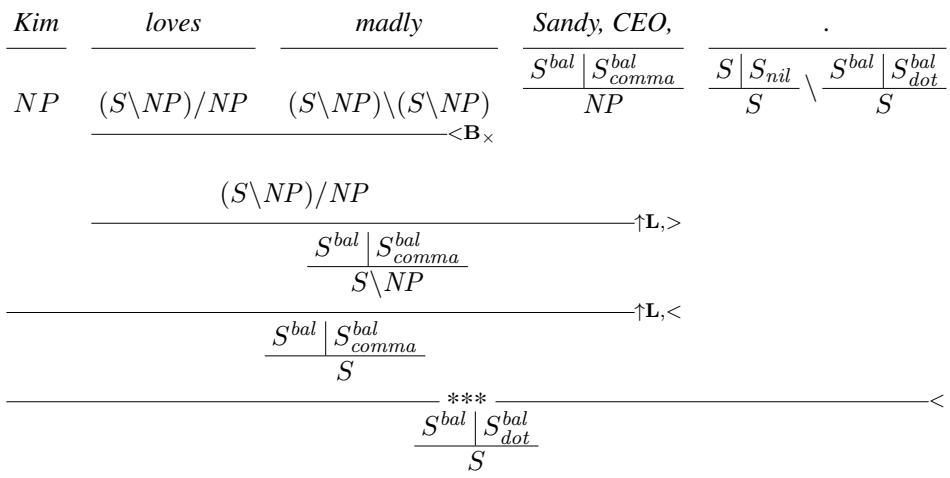


Figure 8: Right Edge Tracking through Crossing Composition in Dynamic Continuized CCG

(3) a. , $\vdash \frac{S \mid S_{nil}}{NP} \backslash \frac{S^{bal} \mid S^{unbal}_{nil}}{NP} / \frac{S \mid S_{nil}}{NP_{pred}}$

b. , $\vdash \frac{S \mid S_{nil}}{NP} \backslash \frac{S^{bal} \mid S^{bal}_{comma}}{NP} / P_{comma} / \frac{S \mid S_{nil}}{NP_{pred}}$

Figure 9: Categories for Unbalanced and Balanced Appositive Commas in Dynamic Continuized CCG

state s with the discourse referent x.

(4) $\lambda s.\{\langle \mathrm{swim}(x), \widehat{sx}\rangle \mid \mathrm{linguist}(x)\}$

More formally, the State.Set monad is defined as in (5). For each type α, the corresponding monadic type $M\alpha$ is a function from states of type s to sets pairing items of type α with such states. The η function injects values into the monad, simply yielding a singleton set consisting of the input item paired with the input state. The *bind* operation $\multimap$ sequences two monadic computations by sequencing the two computations pointwise, feeding each result of m applied to the input state s into π and unioning the results.[13] Less formally, the $\multimap$ operation can be thought of as "run m to determine v in π." Monadic sequencing with bind is what allows the meanings of *a linguist* and *swims* to compose with the indicated result.

(5) State.Set Monad

$$\begin{aligned} M\alpha &= s \to \alpha \times s \to t \\ a^{\eta} &= \lambda s.\{\langle a, s\rangle\} \\ m_v \multimap \pi &= \lambda s.\bigcup_{\langle a,s'\rangle \in ms} \pi[a/v]s' \end{aligned}$$

[13]Note that the notation $m_v \multimap \pi$ is just syntactic sugar for $m \multimap \lambda v.\pi$, which may be more familiar.

Using these notions, the semantics of the comma-delimited appositive phrase *, CEO of XYZ,* is given in (6a) in Figure 10. As the category in (3b) takes a higher-order argument (after combining with the predicative NP and balancing comma), the semantics in (6a) takes a continuation argument k' and returns a continuized meaning (beginning with the second continuized argument k). The continuation argument k' applies the expression $\lambda x \ldots$; consequently, after combining with the continuized meaning of *Sandy*, the constant sandy substitutes for x, as shown in (6b). Via monadic sequencing, the modified noun phrase will then supply the constant sandy as the appropriate argument of the verb, in essentially the same way as it would have had *Sandy* not been modified by the appositive.

Though it's beyond the scope of the paper to go into details, the dynamic appositive semantics proposed here is consistent with Martin's (2016) treatment of supplements, where their typically projective behavior is derived by piggybacking the supplement on the scopal behavior of the modified NP. Moreover, it offers a simplification over Martin's account, as there is no need to appeal to an anaphoric mechanism in order to allow the supple-

(6) a. , *CEO of XYZ,* $\vdash \lambda k'k.k'\lambda xs.\{\langle x, s\rangle \mid \mathrm{ceo}(x, \mathrm{xyz})\}_y \multimap ky$

b. *Sandy, CEO of XYZ,* $\vdash \lambda ks.\{\langle \mathrm{sandy}, s\rangle \mid \mathrm{ceo}(\mathrm{sandy}, \mathrm{xyz})\}_y \multimap ky$

Figure 10: Appositive semantics

mented NP to semantically compose with a verbal predicate.

6 Conclusion

In this paper, we have shown how Combinatory Categorial Grammar's (CCG; Steedman, 2000) flexible treatment of word order and constituency can be problematic for linguistic phenomena where linear order plays a key role. In particular, we have shown for the first time that linear order effects can be problematic for Steedman's (2012) treatment of negative polarity items, and shown that the enhanced control over evaluation order afforded by Continuized CCG (Barker & Shan, 2014) makes it possible to formulate improved analyses of NPIs that account for these effects quite naturally even in a system that combines Steedman's CCG for predicate-argument structure with Barker & Shan's for quantification. In addition, after reviewing how CCG's flexible treatment of word order and constituency are problematic for implementing constraints on balanced punctuation in the style of Briscoe (1994), we have shown how to generalize the approach to encoding evaluation order constraints to properly track punctuation information at the right boundary. As a bonus, we have also taken advantage of Charlow's monadic semantics to implement a novel approach to NP appositives that arguably improves upon Martin's (2016) treatment. Natural next steps for future work include tackling order effects found with binding and crossover as well as exploring the use of machine-learned models to guide the search for derivations.

Acknowledgements

Thanks to Simon Charlow, Dylan Bumford, Jordan Needle, Scott Martin, Mark Steedman and the anonymous reviewers for helpful comments and discussion. This work was supported in part by NSF grant IIS-1319318.

References

Yoav Artzi and Luke Zettlemoyer. 2013. Weakly supervised learning of semantic parsers for mapping instructions to actions. *Transactions of the Association for Computational Linguistics*, 1(1):49–62.

Chris Barker. 2002. Continuations and the nature of quantification. *Natural Language Semantics*, 10(3):211–242.

Chris Barker and Chung-chieh Shan. 2014. *Continuations and Natural Language*. Oxford Studies in Theoretical Linguistics.

Raffaella Bernardi. 2002. *Reasoning with polarity in categorial type logic*. Ph.D. thesis, Utrecht Institute of Linguistics (OTS), Utrecht University.

Ted Briscoe. 1994. Parsing (with) punctuation. Technical report, Xerox, Grenoble, France.

Simon Charlow. 2014. *On the semantics of exceptional scope*. Ph.D. thesis, New York University.

Stephen Clark and James R. Curran. 2007. Wide-Coverage Efficient Statistical Parsing with CCG and Log-Linear Models. *Computational Linguistics*, 33(4):493–552.

Christine Doran. 1998. *Incorporating Punctuation into the Sentence Grammar: A Lexicalized Tree Adjoining Grammar Perspective*. Ph.D. thesis, University of Pennsylvania.

David R. Dowty. 1994. The role of negative polarity and concord marking in natural language reasoning. In *Proceedings from Semantics and Linguistic Theory IV*, Ithaca. Cornell University Press.

Martin Forst and Ronald M. Kaplan. 2006. The importance of precise tokenizing for deep grammars. In *Proc. LREC-06*.

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.

Graham Hutton and Erik Meijer. 1996. Monadic Parser Combinators. Technical Report NOTTCS-TR-96-4, Department of Computer Science, University of Nottingham.

Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2016. Global neural CCG parsing with optimality guarantees. In *Proceedings of the 2016 Conference on*

Empirical Methods in Natural Language Processing, pages 2366–2376, Austin, Texas. Association for Computational Linguistics.

Scott Martin. 2016. Supplemental update. *Semantics and Pragmatics*, 9(5):1–61.

Geoffrey Nunberg. 1990. *The Linguistics of Punctuation*. CSLI Publications, Stanford, CA.

Chung-chieh Shan and Chris Barker. 2006. Explaining crossover and superiority as left-to-right evaluation. *Linguistics and Philosophy*, 29(1):91–134.

Mark Steedman. 2012. *Taking Scope: The Natural Semantics of Quantifiers*. MIT Press, Cambridge, MA, USA.

Michael White. 1995. Presenting punctuation. In *Proceedings of the Fifth European Workshop on Natural Language Generation*, pages 107–125.

Michael White. 2006. Efficient realization of coordinate structures in combinatory categorial grammar. *Research on Language and Computation*, 4(1):39–75.

Michael White, Simon Charlow, Jordan Needle, and Dylan Bumford. 2017. Parsing with Dynamic Continuized CCG. In *Proceedings of the 13th International Workshop on Tree Adjoining Grammars and Related Formalisms*, pages 71–83, Umeå, Sweden. Association for Computational Linguistics.

Michael White and Rajakrishnan Rajkumar. 2008. A more precise analysis of punctuation for broad-coverage surface realization with CCG. In *Coling 2008: Proceedings of the workshop on Grammar Engineering Across Frameworks*, pages 17–24, Manchester, England. Coling 2008 Organizing Committee.