Eliding the Derivation
A Minimalist Formalization of Ellipsis

Gregory M. Kobele

Computation Institute and Linguistics Department, University of Chicago,
Chicago, IL 60637
kobele@uchicago.edu

Summary. In this paper I use the formal framework of minimalist grammars to implement a 'deletion' approach to ellipsis where identity is computed over the derivation. This, in conjunction with canonical analyses of voice phenomena, immediately allows for voice mismatches in verb phrase ellipsis, but not in sluicing. This approach to ellipsis is naturally implemented in a parser by means of threading a state encoding a set of possible antecedent derivation contexts through the derivation tree. Similarities between ellipsis and pronominal resolution are easily stated in these terms. In the context of this implementation, two approaches to ellipsis in the transformational community are naturally seen as equivalent descriptions at different levels: the LF-copying approach to ellipsis resolution is best seen as a description of the parser, whereas the phonological deletion approach a description of the underlying relation between form and meaning.

Keywords: minimalist grammars, deletion, derivational identity

1 Introduction

In the transformational grammar community, analyses of ellipsis which involve reconstructing a syntactic structure have been proposed (Lees, 1960) and reproposed (Chung et al., 1995; Merchant, 2001; Kobele, 2009). The structure reconstructed stands in some, often syntactic, relation to some other syntactic structure, its antecedent. In conjunction with canonical transformational analyses of basic sentence structure, such as a 'phrasal' (as opposed to a 'lexical') approach to passive constructions (Jaeggli, 1986), this kind of approach to ellipsis is able to present a unified theory which neatly captures some differences between elliptical phenomena in the degree to which they are sensitive to syntactic properties of antecedents (Kobele, 2009; Merchant, 2008).

Two mechanisms for dealing with ellipsis are prominent in today's transformational literature: deletion and copying. The first views ellipsis as a process of syntactically conditioned phonological deletion (Merchant, 2001). In this approach, an ellipsis site might be assigned an arbitrarily complex syntactic structure. The second approach, more in line with perspectives in other approaches to grammar, views ellipsis as involving the post-pronunciation substitution of syntactic structure into a syntacticaly atomic empty category (Chung et al., 1995). Both approaches must be complemented with an appropriate 'identity condition,' which allows a phrase to be deleted / inserted just in case it is identical to some other phrase.

Using the formal framework of minimalist grammars (Stabler, 1997), a mildly context-senstitive (Michaelis, 2001) formalization of the minimalist program (Chomsky, 1995), I use the mechanism of deletion to implement a modern version of the 'derivational identity' approach to ellipsis of Lees (1960). Borrowing ideas from Lichte and Kallmeyer (2010), I take phonological deletion to be licensed by exact identity of derivation tree *contexts* (trees with holes at the leaves).

HPSG 2012 Conference/Ellipsis Workshop, July 18-21, 2012, Chungnam National University, Daejeon, Korea

The main advantages of this approach to deletion are (1) that it is naturally implemented in a parser (whose job it is to reconstruct derivation trees), and (2) that in conjunction with a compositional semantics it allows derivation tree contexts to be replaced by their semantic interpretations. Furthermore, in the context of this formalization, the deletion and LF-copying approaches to ellipsis can be viewed as equivalent descriptions at different levels (in the sense of Marr (1982)): the LF-copying approach describes the algorithmic realization of the deletion approach in a parser.

Moving from computation to algorithm, a natural way of implementing the licensing requirement on deletion (that, namely, an identical antecedent be present) involves passing a 'state' containing information about what antecedents are present. This can be managed by using variants of the state monad (Wadler, 1992). In particular, different hypotheses about antecedent availability can be implemented by allowing the state information to flow in different directions (e.g. the state and the reverse state monads). Antecedent choice is made by means of a choice operator, which, as in a continuation-based treatment of pronouns (de Groote, 2006), may be made sensitive to discourse and other factors.

Thus, the formal approach to ellipsis in minimalism outlined here clearly separates various empirical phenomena surrounding ellipsis – factors influencing the choice of antecedent go in the choice operator, restrictions on availability of antecedents are to be accounted for in the state passing mechanism (the implementation of monadic `bind`), and which antecedents exist at all is the provenance of the syntactic analysis.

In the following I first describe informally minimalist grammars, where I explain the notion of derivation tree, and then introduce an operation of deletion. Then I sketch how to implement this approach in a parsing algorithm.

2 Minimalist Grammars

A minimalist grammar has two structure building operations, binary **merge** and unary **move**, whose application to expressions is dependent on the syntactic categories of these expression. Categories are complex, as in categorial grammar, and are structured as lists of atomic *features*, which we will call feature bundles. The currently accessible feature is the feature at the beginning (leftmost) position of the list, which allows for some features being available for checking only after others have been checked. In order for **merge** to apply, the heads of its two arguments must have matching accessible features. These features are eliminated in the derived structure which results from their merger. In the case of **move**, the head of its argument must have an accessible feature matching an accessible feature of the head of one of its subconstituents'. In the result, both features are eliminated. Each feature type has an attractor and an attractee variant (i.e. each feature is either positive or negative), and for two features to match, one must be positive and the other negative. The kinds of features relevant for the **merge** and **move** operations are standardly taken for convenience to be different. For **merge**, the attractee feature is a simple categorial feature, written `x`. There are two kinds of attractor features, `=x` and `x=`, depending on whether the selected expression is to be merged on the right (`=x`) or on the left (`x=`). For **move**, we write `+y` and `-y` for the attractor and attractee feature, respectively. A lexical item is an atomic pairing of form and meaning, along with the syntactic information necessary to specify the distribution of these elements in more complex expressions. We write lexical items using the notation $\langle \sigma, \delta \rangle$, where σ is a lexeme, and δ is a feature bundle. Figure 1 (adapted from Kobele (2006)) gives a small lexicon which derives active and passive sentences like the below.[1]

(1) John will praise Mary

(2) Mary will be praiseed

[1] The 'ϵ' is the empty string, representing a phonetically null element. The hyphen preceding some of the lexemes in the figure indicates that that lexeme is a suffix, and triggers phonological head movement from its complement.

⟨John, d −k⟩	⟨Mary, d −k⟩
⟨praise, =d V⟩	⟨will, =v +k s⟩
⟨−ϵ, =V +k V⟩	⟨−ϵ, =V d= v⟩
⟨−en, =V pass⟩	⟨be, =pass v⟩

Figure 1: A lexicon

2.1 Derivations

A *derivation tree* is a (complete) description of how to construct an expression. Using • to represent the merge operation, and ∘ to represent the move operation, the sentences above have the following derivations:[2]

(3) ∘(•(**will**, •(•(**v**, ∘(•(**agrO**, •(**praise**, **Mary**))))), **John**)))

(4) ∘(•(**will**, •(**be**, •(**-en**, •(**praise**, **Mary**))))))

We can streamline the representation of derivations by moving to a more 'dependency'-like structure, as per the below (derivations 5 and 6 are the streamlined versions of 3 and 4, respectively).

(5) **will**(**v**(**agrO**(**praise**(**Mary**)), **John**))

(6) **will**(**be**(**-en**(**praise**(**Mary**)))))

The derivation trees which are *well-formed*—those which actually represent 'convergent' derivations—form a regular set (Kobele et al., 2007; Mönnich, 2007). This means that they are particularly simple to describe, in contrast to their derived sentences, which do not have a simple direct description. This makes the derivation tree a particularly attractive object from a computational perspective.

3 Deletion

Our deletion operation targets arbitrary *connected* subparts of the derivation tree. We introduce two new operations, **delete**, and **elide**. **Delete** is a lexical operation (i.e. it applies to lexical items, not to arbitrary expressions), and we will write $\ell^\star$ instead of the more cumbersome delete(ℓ), for ℓ a lexical item. The interpretation of the operation **delete** on a lexical item is simply to delete its phonological exponent. This ensures that even discontinuous non-deleted material is treated normally by the grammar. The operation **elide** (which we abbreviate 'E') delimits a stretch of deleted elements as a single eliptical unit, and can be applied only to a derivation tree whose root is deleted. It has no other effect. Derivations 7 and 8 both involve ellipsis at the VP level. Derivation 7 (pronounced as "John will") deletes the entire VP (all elements under the E have a star), and derivation 8 (pronounced as "Mary will be"[3]) deletes just the verb within the VP (only **praise** has a star).

(7) **will**(**v**(**agrO**(E(**praise***(**Mary***))), **John**))

(8) **will**(**be**(**-en**(E(**praise***(**Mary**)))))

A derivation tree with a deleted node can be well-formed only if this node is dominated either by another deleted node or by a node labeled elide. A derivation tree with a node labeled elide can be well-formed only if the expanse of deleted nodes ultimately licensed by this elide node

[2] For readability, I abbreviate lexical items with their stems in **bold**. For the lexical items ⟨−ϵ, =V d= v⟩ and ⟨−ϵ, =V +k V⟩ I write **v** and **agrO** respectively.

[3] The (nonappearance of the) participial morphology is not accounted for yet. I assume the stranded affix filter (Lasnik, 1981), which could be implemented by a realizational approach to morphology in which all paradigm cells of a deleted stem are also empty.

is identical (modulo deletion) to some other part of this derivation tree.[4] For example, the sentences below can be derived (with the additional lexical item $\langle \mathsf{before}, \texttt{=s v= v} \rangle$) – sentence 9 uses derivation 7, 10 and 11 use 8, and 12 uses a derivation like 7 but without the object being deleted.

(9) Bob will kiss Mary before John will

(10) Bob will kiss Mary before Susan will be

(11) Mary will be kissed before Susan will be

(12) Susan will be kissed before John will Mary

4 Computation

The above account of ellipsis is stated at Marr's computational level, which describes what is being computed but not how. The most natural way of implementing the recognition of ellipsis in this context is to separate the *detection* of ellipsis sites from their *resolution*, at least logically (Kobele, 2012a) (although this separation can and should be 'parallelized' on-line). This allows perfectly standard minimalist grammar parsing algorithms (Harkema, 2001) to be used to construct parse trees with unresolved ellipsis sites. Note that because we allow for deletion of contexts, not just subtrees, unresolved ellipsis sites take the form of relation symbols with rank arbitrary (but bound by a function of the length of the sentence).

A theory neutral way of stating an ellipsis resolution algorithm is the following. We are given a type of a *state* and a *selection function* `sel` which determines the best resolution to a particular ellipsis site given a state. We traverse a tree with unresolved ellipsis sites and use `sel` to resolve any ellipsis sites encountered based on the current state.

Two things are of particular interest about this setup. First, this is the same basic framework as the one developed by (de Groote, 2006) for pronoun resolution, and, as noted there, the selection function `sel` can be parameterized by arbitrary contextual information. Thus, the parallels between ellipsis resolution and pronoun resolution (cf. (Hardt, 1993)) are hereby partially explained by treating them in terms of the same (or a similar) mechanism. Second, taking the goal of parsing to be the recovery not of a parse tree but of a semantic interpretation, the state can simply record semantic denotations (paired with syntactic categories) instead of derivation tree contexts. This follows from the existence of a variable free interpretation scheme for minimalist grammars (Kobele, 2012b), which can assign semantic interpretations to arbitrary derivation tree contexts. Thus, although this is an implementation of a syntactic identity theory of ellipsis, we are free to 'leave syntax behind', and work at the level of meanings.

5 Conclusions

This is a formalization of some common themes in the minimalist literature regarding ellipsis. By keeping syntactic identity at the level of the derivation (instead of the derived tree), this allows for some flexibility regarding suface antecedent-ellipsis mismatches. Not all well-known surface mismatches (for example 'vehicle change' (Fiengo and May, 1994)) are so easily dealt with. It is hoped that this formal presentation will serve to make clear the commitments, prospects, and difficulties faced by a deletion under identity theory of ellipsis.

References

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

[4] Cross-sentential ellipsis must here be dealt with by taking discourse and sentence grammar to be identical cf. (Webber, 2004).

Chung, Sandra, William A. Ladusaw, and James McCloskey. 1995. Sluicing and logical form. *Natural Language Semantics*, 3(3):239–282.

Fiengo, Robert and Robert May. 1994. *Indices and Identity*. MIT Press, Cambridge, Massachusetts.

de Groote, Philippe. 2006. Towards a montagovian account of dynamics. In Masayuki Gibson and Jonathan Howell, editors, *Proceedings of SALT 16*, pages 1–16.

Hardt, Daniel. 1993. *Verb Phrase Ellipsis: Form, Meaning, and Processing*. Ph.D. thesis, University of Pennsylvania.

Harkema, Henk. 2001. *Parsing Minimalist Languages*. Ph.D. thesis, University of California, Los Angeles.

Jaeggli, Osvaldo A. 1986. Passive. *Linguistic Inquiry*, 17(4):587–622.

Kobele, Gregory M., Christian Retoré, and Sylvain Salvati. 2007. An automata theoretic approach to minimalism. In James Rogers and Stephan Kepser, editors, *Proceedings of the Workshop Model-Theoretic Syntax at 10; ESSLLI '07*, Dublin.

Kobele, Gregory M. 2006. *Generating Copies: An investigation into structural identity in language and grammar*. Ph.D. thesis, University of California, Los Angeles.

Kobele, Gregory M. 2009 Syntactic identity in survive minimalism: Ellipsis and the derivational identity hypothesis. In Michael T. Putnam, editor, *Towards a purely derivational syntax: Survive-minimalism*. John Benjamins.

Kobele, Gregory M. 2012a. Ellipsis: computation of. *WIREs Cognitive Science*, 3(3):411–418.

Kobele, Gregory M. 2012b. Importing montagovian dynamics into minimalism. In Denis Béchet and Alexandre Dikovsky, editors, *Logical Aspects of Computational Linguistics*, volume 7351 of *Lecture Notes in Computer Science*, pages 103–118, Berlin. Springer.

Lasnik, Howard. 1981. Restricting the theory of transformations: A case study. In Norbert Hornstein and David Lightfoot, editors, *Explanations in Linguistics: The logical problem of language acquisition*, pages 152–173. Longmans, London.

Lees, Robert B. 1960. *The grammar of English nominalizations*. Mouton, The Hague.

Lichte, Timm and Laura Kallmeyer. 2010. Gapping through TAG derivation trees. In *Proceedings of the 10th Conference on Tree Adjoining Grammar and Related Frameworks*, New Haven, CT.

Marr, David. 1982. *Vision*. W. H. Freeman and Company, New York.

Merchant, Jason. 2001. *The Syntax of Silence: Sluicing, Islands, and the Theory of Ellipsis*, volume 1 of *Oxford Studies in Theoretical Linguistics*. Oxford University Press, New York.

Merchant, Jason. 2008. An asymmetry in voice mismatches in VP-ellipsis and pseudogapping. *Linguistic Inquiry*, 39(1):169–179.

Michaelis, Jens. 2001. *On Formal Properties of Minimalist Grammars*. Ph.D. thesis, Universität Potsdam.

Mönnich, Uwe. 2007. Minimalist syntax, multiple regular tree grammars and direction preserving tree transductions. In James Rogers and Stephan Kepser, editors, *Proceedings of the Workshop Model-Theoretic Syntax at 10; ESSLLI '07*, Dublin.

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

Wadler, Philip. 1992. Comprehending monads. *Mathematical Structures in Computer Science*, 2:461–493.

Webber, Bonnie. 2004. D-LTAG: extending lexicalized TAG to discourse. *Cognitive Science*, 28:751–779.