Journal of Language Modelling

VOLUME 5 ISSUE 3
DECEMBER 2017

Institute of Computer Science
Polish Academy of Sciences
Warsaw

Journal of Language Modelling

VOLUME 5 ISSUE 3
DECEMBER 2017

Articles

Tools and resources

JOURNAL OF
LANGUAGE MODELLING

ISSN 2299-8470 (electronic version)
ISSN 2299-856X (printed version)
http://jlm.ipipan.waw.pl/

Published by IPI PAN
Institute of Computer Science, Polish Academy of Sciences
ul. Jana Kazimierza 5, 01-248 Warszawa, Poland

Circulation: 100 + print on demand

Layout designed by Adam Twardoch.
Typeset in XeLaTeX using the typefaces: *Playfair Display*
by Claus Eggers Sørensen, *Charis SIL* by SIL International,
JLM monogram by Łukasz Dziedzic.

EDITORIAL BOARD

Aligning speech and co-speech gesture in a constraint-based grammar

Katya Alahverdzhieva[1]*, Alex Lascarides*[1]*, and Dan Flickinger*[2]
[1] School of Informatics, University of Edinburgh, UK
[2] Center for the Study of Language and Information,
Stanford University, USA

ABSTRACT

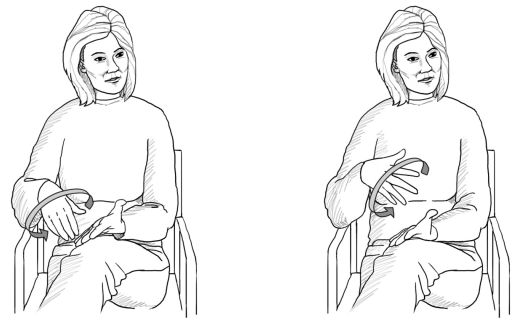
This paper concerns the form-meaning mapping of communicative actions consisting of speech and improvised co-speech gestures. Based on the findings of previous cognitive and computational approaches, we advance a new theory in which this form-meaning mapping is analysed in a constraint-based grammar. Motivated by observations in naturally occurring examples, we propose several construction rules, which use linguistic form, gesture form and their relative timing to constrain the derivation of a single speech-gesture syntax tree, from which a meaning representation can be composed via standard methods for semantic composition. The paper further reports on implementing these speech-gesture construction rules within the English Resource Grammar (Flickinger 2000). Since gestural form often underspecifies its meaning, the logical formulae that are composed via syntax are underspecified so that current models of the semantics/pragmatics interface support the range of possible interpretations of the speech-gesture act in its context of use.

Keywords:
co-speech gesture, constraint-based grammar, compositional semantics, underspecification

1 INTRODUCTION

In face to face conversation, people exchange information via a range of meaningful and visibly accessible communication channels (Goffman 1963); in particular they use "visible bodily actions"

Figure 1:
Gesture depicting mixing mud, example (1)



(Kendon 2004). For instance, in utterance (1),[1] extracted from a conversation where the speaker is describing installing drywall. (Loehr 2004),[2] the speaker performs a circular movement with the right hand over the left palm (see Figure 1) along with the spoken utterance. Both the speech and the hand movement are relevant for the conveyed meaning of mixing mud, and both are produced and perceived as a coherent idea unit (McNeill 1992).

(1) So he mixes [$_N$mud] …

In this article, we analyse signals like (1), in which the hand is spontaneously used to convey meaning in tandem with speech. In the literature, these hand signals are known as *co-speech gesture*, *co-verbal gesture* or *gesticulation* (e.g., Kendon 1972). In *depicting/referential* gestures, the form of the hands visually characterises a salient feature of the referent. The depiction could be *iconic* (McNeill 1992) (e.g., in (1) the hands perform a rotating movement to depict the mud being mixed), or *metaphoric* (McNeill 1992) (e.g., a rotating hand while saying "This was a long, boring process" can designate an iterative process). In *deixis/pointing* gestures, the hand points to a region in space

[1] We adopt the following conventions in utterance transcriptions: the part of the speech signal that is simultaneous with the expressive phase of the gesture, the so-called stroke, is underlined. We include words that start or end at midpoint in relation to the gesture phase boundaries. The pitch accented words are shown in square brackets with the accent type in the left corner: PN (pre-nuclear), NN (non-nuclear) and N (nuclear).

[2] For this and for all subsequent examples that are cited as Loehr (2004), we are grateful to Daniel Loehr who kindly provided us with an annotated corpus of speech and co-speech gesture. We used this corpus to study depicting gestures.

so as to identify the referent's location in Euclidean space. The pointing can be *concrete* (McNeill 1992), as when pointing to something that's physically present in the communicative situation. It can also be *abstract* (McNeill 1992): the referent is a virtually created object in the gesture space just in front of the speaker, and its location in the gesture space constrains its physical location; e.g., a speaker, while describing her apartment that's on the other side of town, extends her right hand to the right periphery while saying "The bedroom is on the right". Formless flicks of the hand, beating the time along with the rhythm of the speech are known as *beats*. The current analysis focusses on depicting and pointing co-speech gestures.

We adhere to current theories of gesture (Cassell *et al.* 1999; Lascarides and Stone 2009a; Pfeiffer *et al.* 2013), in that we assume that co-speech gesture can affect the truth-conditional content of the speech-and-gesture action. Both deictic gestures and iconic representations say something about the world and as such they have propositional content; this extends to pictorial representations as well (Abusch 2014; Grzankowski 2015).

Our paper contributes to the existing approaches to integrating the contents of speech of co-speech gesture in a single semantic unit (McNeill 1992; Kendon 2004; Bavelas and Chovil 2006; Engle 2000; Giorgolo 2012) in that we explore the coordination patterns of the two modalities, we formalise them within an integrated grammar, and we spell out the gesture's semantic contributions to the proposition that is conveyed by the speech-gesture action. The main challenges are two-fold: on the one hand, the gesture signal is massively ambiguous (Lascarides and Stone 2009a); on the other, the speech-gesture integration is not a free-for-all, in that the *form* of the speech-gesture action rules out certain interpretations of it, whatever its context of use. To illustrate gesture's ambiguity, consider again the hand movement in (1). Taken out of its speech context, this gesture could be a depiction of a circular movement (e.g., the turning of a wheel), or it could refer to the object being rotated (e.g., the wheel itself), or it could refer to an iterative process. It is only via context that gesture receives a specific meaning: the content conveyed by the rotating movement while saying "He mixes mud" is distinct from that while saying "It's a huge, long boring process".

The form of a deictic gesture is also imprecise on the region pointed out by the hand and what is being designated (Kühnlein *et al.* 2002): when pointing in the direction of a book with an extended index finger, does the deictic gesture identify the physical object book, the book's content, or the location of the book – e.g., the table?

This ambiguity notwithstanding, the form of the gesture, abstracted away from its context of use, conveys some meaning, no matter how incomplete it might be. A depicting gesture, by the definition of iconicity, must support a perceptual resemblance between the gesture's form and its denotation (Kendon 2004; Kopp *et al.* 2007): i.e., the gesture's movement, hand shape etc. visualise qualitative characteristics of the referent. Deixis, on the other hand, indexes spatial reference in Euclidean space by projecting the hand to a region that is proximal or distal in relation to the speaker's location (e.g., Levinson 1983). Through deictic gestures, people anchor the referents in their utterances to the physical context (Kaplan 1989). This difference between depicting gestures and deictic gestures is accounted for in how we model the form-meaning mapping, and we also support the analysis of gestures that are *both* deictic and depictive simultaneously (and so inherit the characteristics of both gestural types).

Outline

This article is structured as follows: in Section 2, we discuss the ambiguous form-meaning mappings of the speech-and-gesture signal, assuming a coherence-based pragmatic theory. In Section 3, we introduce examples to motivate a grammar-based approach to co-speech gesture. We then proceed with a discussion of related work and our distinct contribution (Section 4). In Section 5, we discuss how to formally represent gesture form and map this form to (underspecified) meaning. In Section 6, we propose domain-independent grammar rules which are based on the empirically extracted generalisations. Section 7 reports on the grammar implementation and evaluation.

2 AMBIGUOUS FORM–MEANING MAPPING

There is a balance to be struck between constraining the mapping from form to meaning, while ensuring that existing pragmatic theories will

support inferring the context-specific interpretations from the underspecified meanings derived only from form. The aim of this section is to use examples of speech-gesture actions to motivate one way of striking that balance. We first introduce an existing coherence-based model of pragmatics, which we assume underlies the inferences from the meaning that is derived from form alone to a preferred pragmatic interpretation in context. We then use this to motivate speech-gesture attachment ambiguities by illustrating how each syntax tree supports a different interpretation of the speech-and-gesture action, given the assumed pragmatics model. We also argue that licensed attachments are constrained, despite the multiple ways co-speech gestures can relate to speech.

2.1 *Pragmatic theory background*

In this paper, we assume a coherence-based model of the semantics/pragmatics interface as discussed in the literature of discourse interpretation (e.g., Hobbs 1985, Kehler 2002). The main principle of a coherence-based pragmatic theory is that discourse content is dependent on *coherence relations* – e.g., Elaboration, Explanation, Contrast, Contiguity – which link the meaning of its segments together. Identifying coherence relations is a defeasible process, informed by the compositional and lexical semantics of the units and contextual information such as real-world knowledge.

For instance, the pragmatic interpretation of the discourse in (2) involves the following contents: Max fell, John pushed Max, and the latter explains the former (so the pushing caused the falling and hence preceded it).

(2) Max fell. John pushed him.

Using the notation of Segmented Discourse Representation Theory (SDRT, Asher and Lascarides 2003), as shown in (3), this is represented as a rooted hierarchical set of labels – each label corresponds to a discourse segment – with each label associated with some content: π_1 is associated with the content that the event e_1 of Max m falling happened before now; segment π_2 with the content that the event e_2 of John j pushing x, where x is identical to m, happened before now; and the (root) segment π_0 stipulates that π_2 explains π_1 (in other words, the content of π_2 explains why the content of π_1 is true).

(3) $\pi_0 : Explanation(\pi_1, \pi_2)$
$\pi_1 : fall(e_1, m) \wedge e_1 < now$
$\pi_2 : push(e_2, j, x) \wedge x = m \wedge e_2 < now$

The linguistic grammar doesn't identify the antecedent m to the pronoun x. Rather, "him" introduces an *underspecified* equality condition between the newly introduced referent x and some antecedent – written $x =?$. Generally, (disambiguated) linguistic form yields an Underspecified Logical Form (ULF), because syntax on its own does not fully resolve all semantic and anaphoric ambiguities. Similarly, the grammar does not introduce the Explanation relation between the segments. Rather, identifying this coherence relation and the antecedent m to x (thereby replacing $x = ?$ with $x = m$ in the logical form of the discourse) is achieved via commonsense reasoning, using the ULFs of the clauses as premises. Moreover, the assumption that π_2 is coherently related to π_1 is what makes m an available antecedent for x.

Following Lascarides and Stone (2009a), we assume that gestures are elementary discourse units (that is, segments at the leaves of the hierarchical discourse structure); so interpreting gesture involves inferring coherence relation(s) between it and other speech units and gesture units. Furthermore, Lascarides and Stone (2009a) stipulate that co-speech gesture *must* be coherently related to its synchronous speech, and it *can* be related to other units as well. The main aim of this paper is to model this *necessary* connection between co-speech gesture and its synchronous speech. In line with theories of dynamic semantics and discourse interpretation (Hobbs 1985; Kehler 2002; Asher and Lascarides 2003), we further assume that there are constraints on which antecedents are available for resolving the anaphoric elements of the current discourse unit. In speech-only discourse, antecedents to anaphora in the discourse unit π must be introduced in π itself or in a unit π' that π is coherently related to. Following Lascarides and Stone (2009a), we carry over these constraints to gesture: i.e., *all* individuals that are a part of the pragmatic interpretation of a gesture behave like anaphoric expressions – they must bind via a bridging relation to an available antecedent (Asher and Lascarides 1998). Thus inferring a pragmatic interpretation of gesture is dependent on inferring how it coherently connects to available speech unit(s).

The meaning representations that we derive from the form of a sentence with co-speech gesture must respect the above constraints on interpretation. To achieve this, we make the choices of speech and gesture integration – which we formally express by attachments in the syntax tree – determine the speech phrase that the gesture is coherently related to. This in turn affects which referents, introduced in speech, are available antecedents for resolving the underspecified gesture meaning (given just its form).

Lascarides and Stone (2009a) observe additional constraints on antecedents for resolving gesture interpretation; constraints that we assume here. Specifically, they claim that the antecedent for resolving gesture can be introduced by a gesture or a linguistic discourse unit, but antecedents for resolving linguistic anaphora cannot be introduced by depicting gestures. This doesn't apply to deixis: a linguistic anaphor can co-refer with a referent that's pointed at. For instance, when a person points at a knife and says "It's sharp", it is perfectly acceptable for "it" to refer to the knife introduced by the deictic gesture. In contrast, when a person says "He cut the cake" and makes a 'cutting' gesture with a vertically flat palm to depict the instrument used for cutting, it is rather unnatural to continue this discourse with "It was sharp" where "it" refers to the knife introduced by the iconic gesture.

By drawing on standard methods from formal linguistics, our goal is to make the analysis of a discourse featuring co-speech gestures compatible with the analysis of purely linguistic discourse. Given the fact that we are adopting a coherence-based theory, the pragmatic interpretation of co-speech gesture is dependent on the content of the linguistic signal it is coherently related to. With this in mind, we introduce the notion of *speech-gesture alignment* to roughly designate: (i) that speech and gesture are coherently related; and (ii) that resolving the (underspecified) semantics of gesture to a specific interpretation and inferring a coherence relation are logically co-dependent tasks. We shall refine the notion of alignment in Section 3.3 after a discussion of how linguistic form and gestural form, including their relative timings, constrain the alignment configurations. In the next section, we illustrate the various ways in which a gesture can be interpreted in context.

2.2 *Ambiguous form-meaning mapping*

Syntactic attachment ambiguities and semantic scope ambiguities are ubiquitous in grammars. For instance there is the non-unique choice for attaching the PP in "John saw the man with the telescope". And there's the non-unique semantic scope of the quantifier in "every dog probably did not walk" – "probably" semantically outsopes the negation, which outscopes "walk", but the quantifier "every man" may outscope "probably", or have narrow scope to "probably" but outscope the negation, or have narrow scope to the negation. Most grammars have to handle semantic scope ambiguity in the absence of syntactic ambiguity.[3] So syntax derives a ULF that underspecifies semantic scope.

We will now argue that the range of plausible pragmatic interpretations of co-speech gesture can likewise be analysed via a non-unique choice of attachment of the co-speech gesture to speech and a non-unique way of resolving scope in the ULF that gets composed via such attachments. In essence, these sources of ambiguity familiar from linguistics can also capture ambiguities in co-speech gestures. In Section 3.1, we will then argue that not only *can* one model co-speech gesture ambiguity this way, but one *should*.

We use a slight modification of example (1), namely (4), to discuss the ambiguous form-meaning mapping of depicting gestures. Its plausible pragmatic interpretations are presented in SDRT notation, except that we ignore tense and presupposition, and (following the English Resource Grammar (ERG, Flickinger 2000)), events are not existentially bound.

(4) John mixes mud
Same gesture as in (1)

Intuitively, one of the possible denotations of the circular hand movement is paraphrasable as "the mud is going round in horizontal circles". This interpretation is regimented in the LF in (5), which features an Elaboration relation between the speech content $mud(x)$ (labelled π_s) and the gesture content labelled π_g – a horizontal rotating event e' over a substance x' that is made equal to the 'mud'

[3] For instance, CCG (Steedman 2000) and Montague Grammar (Montague 1988).

referent x introduced in π_s. The speech-gesture action conveys "John mixes mud, (specifically) the mud that is going round". Like (2), this LF consists of a hierarchical structure of coherently related segments.

(5) $\pi_s : mud(x)$
$\pi_g : \exists x'(substance(x') \wedge rotate(e', x') \wedge horizontal\_motion(e'', e')$
$\wedge\, x = x')$
$\pi_0 : \exists x(john(j) \wedge mix(e, j, x) \wedge Elaboration(\pi_s, \pi_g))$

The constraints on anaphoric reference imposed by the discourse structure in (5) license using x as an antecedent for specifying the content of π_g (Asher and Lascarides 2003; Lascarides and Stone 2009b): x is available because it's 'introduced' by the predication $mud(x)$ – or more precisely, using HPSG terminology, x is the semantic index of $mud(x)$ (its first argument which introduces a noun variable) – and $mud(x)$ is a part of π_s, to which π_g is coherently related.

Further, this LF represents one way of resolving the underspecified semantic scope of the ULF that you would get by attaching the gesture to the NP "mud" in the syntax tree. Specifically, following the standard approach to semantic composition (Sag and Wasow 1999; Copestake *et al.* 2001), assume the semantic component of the construction rule that attaches gesture to a linguistic unit introduces an (underspecified) *coherence relation* – here resolved to Elaboration – between the gesture and the predications in that linguistic unit, but the ULF so derived underspecifies the relative scope of this (underspecified) coherence relation and the quantifiers in the linguistic unit. Then the ULF derived by attaching the gesture to the NP "mud" would force the coherence relation to outscope the predicate $mud(x)$ but it won't outscope the predicates $mixes(e, j, x)$ or $john(j)$. Proposition (5) is a fully specific logical form that is licensed by this ULF. Here, $\exists x$ *must* outscope the coherence relation because free occurrences of x are forbidden (Copestake *et al.* 2005).

An alternative pragmatic interpretation of the co-speech gesture in (4) is that it depicts the event of mud going round as a *result* of the mixing. A formal rendition of this interpretation is given in (6).

(6) $\pi_s : \exists x(mud(x) \wedge mix(e, j, x))$
$\pi_g : \exists x'(substance(x') \wedge rotate(e', x') \wedge$
$\qquad horizontal\_motion(e'', e') \wedge x = x' \wedge cause(e, e'))$
$\pi_0 : john(j) \wedge Result(\pi_g, \pi_s)$

Unlike (5), the gesture qualifies the event e of mixing – e is available because it's the semantic index of $mix(e, j, x)$, which is a part of π_s. Here, the speech content π_s and the gesture content π_g are coherently related via Result (rather than Elaboration): a rough linguistic paraphrase would be "By making it go round, John was mixing mud". In essense, the gesture here functions roughly like a free adjunct.

This interpretation can be derived by attaching the gesture to a linguistic unit whose timing is (again) not *equal* to the timing of the gesture (though they temporally overlap), and then resolving the ULF that results from this attachment to a fully specific logical form. Here, (6) can be derived from the ULF you get by attaching the gesture to the VP "mixes mud": this attachment forces π_s to include the predication $mix(e, j, x)$. Consequently, the quantifier $\exists x$ can now have narrower scope than the coherence relation, as shown. This contrasts with attachment to the NP "mud": this attachment ruled out $mix(e, j, x)$, and hence also $\exists x$, from being within the scope of the coherence relation. Further, since the predication $john(j)$ in (6) isn't a part of π_s, j is not available for resolving the content of π_g.

The particular linguistic grammar that we use in this paper to analyse co-speech gesture – specifically the ERG (Flickinger 2000) – makes the ULF generated by VP attachment the same as that derived by S attachment. For example, the adverbial in *Probably John mixed mud* and *John probably mixed mud* attaches to the S and VP nodes respectively, but in both cases the ULF forces the modal introduced by *probably* to outscope $mixes(e, j, x)$ and it *underspecifies* whether it also outscopes $john(j)$ and/or $mud(x)$, or not. Thus (6) is also derivable from the ULF you get by attaching the gesture to the S node. An alternative fully scoped form of this ULF corresponds to a further plausible interpretation of the gesture:

(7) $\pi_s : \exists x(john(j) \wedge mud(x) \wedge mix(e,j,x))$
$\pi_g : \exists x'(agent(j') \wedge substance(x') \wedge rotate(e', j', x') \wedge$
$\quad horizontal\_motion(e'', e') \wedge x = x' \wedge e = e' \wedge j = j')$
$\pi_0 : Depiction(\pi_s, \pi_g)$

Unlike (5) and (6), $john(j)$ is now outscoped by the coherence relation; so j is available for resolving the content of π_g. As before, the choice of antecedents for specifying the content of π_g interacts with the choice of coherence relation: here, the coherence relation is Depiction and

the overall content is roughly paraphrasable as another free adjunct: "As he was making it go round, John was mixing mud".

The interpretations in (5), (6) and (7) all feature identity between a referent introduced by the co-speech gesture and a referent introduced by speech. However in (8) the gesture does not denote a salient property of the referents introduced in speech: instead, it qualifies the speech act of questioning (signalled by a rising intonation). A rough paraphrase of the meaning of the multimodal action in (8) would be "Are you telling me that John mixes mud?". Interpreting the gesture in this metaphorical way (see the LF in (9)), and inferring a Metatalk relation (Polanyi 1985) whose semantics is defined in terms of the *speech act* rather than the domain-level content, would be supported via an attachment of the co-speech gesture to the S node.

(8) John mixes mud?
Speaker's right hand is vertically open with palm facing up. The speaker moves it forward to the frontal space.

(9) $\pi_s : question(\exists x(john(j) \wedge mud(x) \wedge mix(e,j,x)))$
$\pi_g : question(tell(e',you,p) \wedge p = \pi_s)$
$\pi_0 : Metatalk(\pi_s, \pi_g)$

While the attachments we've proposed deviate from McNeill's (1992) claim that co-speech gesture is semantically related to its *temporally simultaneous* speech phrase, we remain agnostic about his claims (and those of others) about the underlying production processes – e.g., McNeill's claim that decisions about which contents are expressed in which channel stem from a single (complex) thought.

3 SPEECH–GESTURE ALIGNMENT AS SHOWN IN DATA

This section introduces examples of speech-gesture actions that illustrate that despite their ambiguities, speech-gesture alignment is jointly constrained by prosody, linguistic syntax and relative timing of speech and co-speech gesture. This serves as qualitative evidence for: (a) encoding the constraints on speech-gesture alignment within a grammar (rather than entirely via pragmatics); and in particular (b) suitably constraining the application of construction rules of the kind we

described in the prior section. The examples we use as evidence include both constructed examples (to illustrate our judgements about ill-formedness) and examples extracted from existing corpora.

3.1 *Speech-gesture alignment and prosody*

We begin with the constructed example (10), which reflects intuitions of native speakers about multimodal grammaticality.

(10) * Your [$_N$mother] called.
The speaker puts his hand to the ear to imitate holding a receiver.

Intuitively, it seems anomalous to perform the gesture along the unaccented "called", even though the gesturing hand is shaped as holding a receiver and can thus be associated with calling. This anomaly would not arise if the gesture was performed along the whole utterance (or a part of it) which, importantly, includes the prosodically prominent element "mother": e.g., "mother called" or "your mother called". As suggested by Mark Steedman (personal communication), gestures exhibit contrastive properties in analogy to those conveyed by pitch accents. If this is so, then it's not surprising if a co-speech gesture is well-formed only if, unlike (10), it temporally overlaps with a contrastive component that's signalled via prosodic prominence (this is not to say that gesture performance is *driven* by prosody, but rather that their performances are mutually constraining). Further, a pragmatic interpretation where the gesture depicts calling must be sourced in a syntactic derivation where the gesture is aligned with a linguistic unit that includes "called" – prosody constrains the gesture to be aligned with a phrase that includes "mother", but the event of calling is available to its interpretation only if it aligns with a phrase that includes "called" as well. Thus, just like with purely linguistic discourse, considerations about plausible pragmatic interpretations can serve to resolve syntactic ambiguities that are licensed by the construction rules in the grammar. Further, this strong relationship in (10) between the performance of the gesture and prosody is in line with the empirical findings of Giorgolo and Verstraten (2008), who isolated prosody as the parameter that influences the perception of multimodal well-formedness vs. multimodal ill-formedness.

Considering that form (here, prosody) constrains what part of the speech signal a co-speech gesture can align with, we define align-

Figure 2:
Gesture depicting “greasy”, example (11) (Kendon 2004)

ment as a constraint on grammaticality. Ungrammatical (and hence misaligned) speech and co-speech gestures comprise cases where the timing of co-speech gesture relative to the timing of speech does not validate *any* construction rule in the grammar by which speech and gesture may be combined; and our aim is to ensure that such constraints on the construction rules match native speakers’ judgements about ill-formedness.

3.2 *Speech-gesture alignment and syntax*

To illustrate that linguistic syntax influences decisions about which phrase a co-speech gesture semantically aligns with, consider utterance (11), where the speaker is discussing new owners of a factory finding it filthy. Along with “greasy…”, the speaker’s hands spread out to the left and right periphery (Figure 2) so as to designate some spatial extent, some closed area being made greasy (Kendon 2004).

(11) First of all they made [pause 0.1 sec] everything
[$_N$* gre]asy in the whole room place.

Consider how moving the timing of this gesture affects its meaning. If the gesture onset was moved a few milliseconds earlier so that it happened along “made everything greasy” or if it was held further so as to span “made everything greasy in the whole room”, this would not change the interpretation of it: it still designates an enclosed area that’s greasy. This interpretation would also remain unchanged if the primary pitch accent were on “everything” rather than “greasy”, and the gesture temporally coincided with “everything”. However, the gesture cannot receive this interpretation if it temporally coincides only with the subject NP “they” (which in turn would need to be accented for the speech-gesture action to be well-formed): now it designates

a spatial referent for "they" in the gestural space, and cannot qualify the spatial extent of greasiness. These variations suggest that a gesture that temporally coincides with "they" can only semantically align with "they", but a gesture temporally coinciding with any element in a VP can semantically align with the VP, sub-portions of the VP containing the temporally coinciding words, and with the whole clause.

A special class of deictic gestures behave differently with regards to the semantic effects of prosody and timing, however. In (12) from the annotated AMI corpus (Carletta 2007), the deictic gesture is performed along with the prominent "Thank you" but its denotation binds to that of the NP "the mouse". The alternative interpretation where the gesture signal and the speech signal are bound through a causal relationship – i.e., handing the mouse is the reason for thanking the addressee – is not possible, since it's clear in context that "Thank you" is related to what came in the *previous* discourse (i.e., projecting the presentation in slide show mode in response to the speaker's request).

(12) [$_N$Thank] you. [$_{NN}$I'll] take the [$_N$mouse]
Speaker's right hand is loosely open, index finger is loosely extended, pointing at the computer mouse.

In (13) (again from the AMI corpus), the deixis happens along the nuclear accent "said", but it identifies the individual that resolves the pronoun "she" coming from speech.

(13) And a as she [$_N$said], it's an environmentally friendly uh material
The speaker extends her arm with a loosely open palm towards the participant seated diagonally from the speaker.

In these examples, the gesture would fail to map to the intended meaning if the grammar were to license attaching a co-speech gesture only to its temporally simultaneous linguistic phrase.

Based on Lascarides and Stone (2009a), we formalise the location of the pointing hand with the constant $\vec{c}$; this marks the physical location of the tip of the index finger. This combines with the features of the pointing hand – the hand shape, the orientation of the palm and fingers, and the hand movement – to determine the spatial region $\vec{p}$ that's designated by the gesture – e.g., a stroke with an extended

index finger will make $\vec{p}$ a line (or a cone) that starts at $\vec{c}$ and continues in the direction of the index finger. Abstract deixis identifies referents that are not physically salient in the communicative situation. To account for this inequality between the gestured space and actual denotation, Lascarides and Stone (2009a) use the function v to map the physical space $\vec{p}$ designated by the gesture to the space $v(\vec{p})$ it denotes (and they claim that the value of v is pragmatically determined). Essentially, $\vec{p}$ is not equal to $v(\vec{p})$ in cases where the referent introduced in the gesture space is not physically present. Conversely, $\vec{p}$ equals $v(\vec{p})$ when the referent introduced by the gesture is at the physical coordinates identified in the gesture space.

With this in mind, we observed in all the annotated corpora we examined[4] that the temporal/prosodic mismatch occurred only in cases where the visible space $\vec{p}$ designated by the gesture was *equal* to the space $v(\vec{p})$ it denoted, i.e., the function v that maps the space identified by gesture to the actually denoted space resolves to equality. So we shall capture this finding in the grammar via a construction rule that allows gesture to align with a spoken word that is not prosodically marked and/or that doesn't temporally overlap with the gesture, but only if the deictic referent is physically located at the exact coordinates identified by the pointing hand.

Bearing in mind that we are restricting our study and analysis to only those gestures that temporally overlap with speech (i.e., co-speech gestures), these examples provide evidence that their semantic alignment depends on the syntax and prosody of the speech signal, as well as the relative timing of the gesture and speech. This motivates encoding the constraints on alignment *within a grammar*, for this is where information about syntactic constituency is expressed. The alternative approach would be to infer speech-gesture alignment at the pragmatic level, via the commonsense reasoning that resides there for inferring which discourse units are coherently connected to which other units. But this alternative is incompatible with existing and well-established assumptions about the interface between syn-

[4] To study depicting gestures, we used a 165-second collection of four recorded meetings, annotated for gesture events and intonation events in the ToBI framework (Loehr 2004). To study deictic gestures, we used two multimodal corpora: a 5.53 min recording from the Talkbank Data,[5] and observation IS1008c, speaker C from the AMI corpus (Carletta 2006).[6]

tax, semantics and pragmatics. For instance, our discussion of example (11) showed that the temporal relationship between subject NP/VP boundary and the gesture profoundly affect the possible interpretations. To capture this fact, pragmatics would need access to the *syntax* of the speech. However, there is no formal model of pragmatics that supports that kind of architecture, without pragmatics being fully integrated into the grammar itself along the lines of Dynamic Syntax (Kempson *et al.* 2000). In contrast to the non-modular approach of Dynamic Syntax, we aim to maintain a conservative, well-established and modularised interface between syntax, semantics and pragmatics, so that implementations of our grammar can be supported by standard methods for computing discourse meanings (e.g., statistical discourse parsers, Afantenos *et al.* 2015).

Accordingly, we will develop a speech-gesture grammar using standard techniques for syntactic derivation and semantic composition, where the constraints on attaching co-speech gesture to a linguistic constituent are defined in terms of relative timing, prosody and linguistic syntax.

The examples we've discussed so far motivate allowing attachments of gesture to linguistic constituents whose timing is *not* identical to the timing of the gesture; we saw in Section 2.2 that making alignment equivalent to temporal simultaneity would under-generate the range of plausible pragmatic interpretations. Rather, the choices of attachment, and hence ultimately the choices of what the gesture means, are determined by the prosodic properties and constituent boundaries of the speech signal as well as relative timing.

3.3 *Speech-gesture alignment*

Given our assumptions about constrained inference in pragmatics, and also given our observations of how form affects the speech-gesture interaction, we now refine the notion of alignment as follows:

Definition 1 (Speech-gesture alignment)**.** *Our choice of which speech phrase a gesture (stroke) can align with is guided by the following factors:*

i. the final interpretation of the gesture in specific context of use;

ii. the speech phrase whose content is semantically related to that of the gesture given the value of (i); and

iii. the syntactic structure that, with standard semantic composition rules, would yield a ULF *supporting (i) and hence also (ii).*

The derivation of the single speech-gesture syntactic structure, which is constrained by the prosody of the temporally overlapping speech signal, is achieved within the grammar. This definition encompasses both form (introduced in clause (iii)) and meaning (all three clauses). We capture semantic alignment of speech and gesture via attachment in a single syntax derivation tree, because – as shown – syntax (among other things) governs semantic alignment. If there is a choice as to which phrase a co-speech gesture can align to, then this is modelled via a combination of structural – i.e., attachment – ambiguity and semantic scope ambiguity that's licensed by the ULF so-derived. The semantic effects of alignment are thus captured using standard methods of semantic composition on the derivation tree. Given the theory of pragmatics we aim to support, the construction rules combining speech and a depicting gesture introduce an (underspecified) semantic relation $vis\_rel(s,g)$ (visualising relation) between the content g of the depicting gesture and the content s of the speech constituent to which the gesture attaches, which captures the fact that speech and gesture are coherently connected (Lascarides and Stone 2009a). The (underspecified) relation that's introduced by the construction rules that combine deixis and speech is $deictic\_rel(s,g)$ (Lascarides and Stone 2009a). The resolution of these underspecified relations to a pragmatically preferred and specific value happens externally to the grammar at the semantics/pragmatics interface.[7] In Section 6 we discuss the formal framework and in Section 7 the implementation in HPSG.

4 PREVIOUS WORK AND CONTRIBUTION

This paper aims to demonstrate that informal observations about the relationship between speech-gesture form and meaning can be regimented formally, using standard techniques from linguistics. In par-

[7] Resolving the underspecified relations is a matter of commonsense reasoning which includes the underspecified semantics produced by the grammar, as well as real-world knowledge. A relation such *vis_rel* is a supertype of the more specific Depiction and Result.

ticular, we use standard techniques for deriving logical form from a syntax tree within a grammar, while ensuring that the meaning representations so derived comply with the requirements imposed by existing formal models of pragmatics.

The idea of integrating speech and gesture within a grammar is by no means new, with several such proposals established over the past 20 years (see, *inter aliae*, Johnston 1998a,b, Kühnlein *et al.* 2002, Paggio and Navarretta 2009, Giorgolo and Asudeh 2011). Further, the "constituent structure" of gesture, as well as its syntactic function for the integration within the language, has also been a matter of research (see Fricke 2008, Müller *et al.* 2013). And the construction of meaning across speech and gesture has been the subject of analysis within construction grammars (Steen 2013).

But there are a few main differences between this prior work and our approach. First, we claim that the speech phrase that gesture aligns with is not determined uniquely by when the gesture was performed. Whilst the TIME feature matters, we also constrain alignment via prosody and syntactic notions such as headedness. Further, in contrast to these prior grammars, we aim for a *domain independent* analysis, and so we must fully capture all linguistically licensed semantic alignments between speech and co-speech gesture, rather than only those that are plausible in the chosen domain of application. The other main difference lies in the semantic component of the grammar. In particular, we draw on recent advances in deriving an Underspecified Logical Formula (ULF), which allows the grammar developer to capture semantic ambiguity in the absence of syntactic ambiguity. The above grammatical approaches all assume that every semantic ambiguity corresponds to a syntactic ambiguity.

There are previous semantic analyses of gesture (Lücking *et al.* 2006b; Lascarides and Stone 2009a) that assume a grammar produces an underspecified meaning representation: these theories focus on how contextual information contributes to mapping the underspecified meaning that's derived from form into a fully specific and pragmatically preferred interpretation. Our work contributes to this by providing a grammar framework that produces the form-meaning mappings they assume. In doing so, we not only capture informal observations about gestural ambiguity, but our formal model uses well-established methods from linguistics to produce a meaning

representation that is compliant with current models for multimodal processing at the semantics/pragmatics interface.

To achieve that, we perform two dependent tasks: first, we extract generalisations from the existing literature and from our own observations in annotated multimodal corpora about the syntactic and semantic well-formedness of speech-gesture signals; second, we use the extracted generalisations to define a precise grammar that models the form of the speech, the form of the gesture and the form of their combination, producing ULFs of speech and gesture using standard methods of syntactic derivation and semantic composition from linguistics. We also demonstrate that the grammar can be implemented by extending an existing linguistic grammar.

5 MAPPING GESTURE FORM TO MEANING

5.1 *Modelling gesture form*

One major difference between speech and gesture is how the meaning gets derived from the form of the signal. Gestures are 'global' and 'synthetic' (McNeill 1992), i.e., the meanings of the various features of a gesture's form – such as the direction of the movement, the hand shape, the location of the hands, etc. – determine the meaning of the gesture as a whole. This is unlike the semantic compositionality via natural language syntax. Following previous work (Kopp *et al.* 2004, Lascarides and Stone 2006, Hahn and Rieser 2010, among others), we regiment this difference by using Typed Feature Structures (TFS) since they support a *non-hierarchical* representation of the distinct aspects of the gesture's form. The gesture type designates its category: e.g., *depict-literal* for literally depicting gestures (Figure 3) and *deictic-abstract* for abstract deixis (Figure 4), of the kind exhibited in (14):

(14) I [$_{PN}$enter] my [$_{N}$apartment]
Speaker's hands are in centre, palms are open vertically, finger tips point upward; along with "enter" they move briskly downwards, after the downward move, the palms are still vertically open but this time the finger tips point forward.

The feature-value pairs of a depicting gesture capture every aspect of the form of the hand that (potentially) contributes to its meaning: the hand shape, the orientation of the palm and fingers, the location

Figure 3:
TFS representation of the form of the depicting gesture in (1)

depict-literal	
HAND-SHAPE	bent
PALM-ORIENT	towards-down
FINGER-ORIENT	towards-down
HAND-LOCATION	lower-periphery
HAND-MOVEMENT	circular

Figure 4:
TFS representation of the form of the deictic gesture in (14)

deictic-abstract	
HAND-SHAPE	flat
PALM-ORIENT	towards-centre
FINGER-ORIENT	away-body
HAND-MOVEMENT	down
HAND-LOCATION	$\vec{c}$

of the hand relative to the speaker's torso and the hand movement. With deictic gestures, the shape of the hand determines the region of space that is identified by the pointing hand: e.g., an extended index finger identifies a line or a cone that starts from the tip of the index finger; with a vertical open hand, the designated region is a plane. Recording the form of the pointing hand is essential, because prior work shows that it is significant for interpreting its meaning in context (Kendon 2004): e.g., an extended index finger typically singles out an individuated object while a vertical open hand typically denotes a *class* of objects rather than an individuated object, or it serves a pragmatic function such as offering the floor or citing someone else's contribution to the discourse. The hand location of a deictic gesture is represented via the constant $\vec{c}$. This, combined with the deixis form features, determines the region $\vec{p}$ actually marked by the gesture.

5.2 *Modelling meaning*

As we've already highlighted, a well-established method for handling cases where form does not fully determine meaning is semantic underspecification. All frameworks for semantic underspecification – e.g., Quasi-Logical Form (Alshawi 1992), Underspecified Discourse Representation Theory (Reyle 1993), the Constraint Language for Lambda Structures (Egg *et al.* 2001), Hole Semantics (Bos 2004), Minimal Recursion Semantics (Copestake *et al.* 2005), Regular Tree Grammars (Koller *et al.* 2008) – construct from a fully disambiguated form an abstract representation of meaning that can resolve to several distinct specific messages in context, rather than deriving those specific representations from syntax directly, and assuming a syntactic ambiguity

for every semantic ambiguity. Technically, the ULF derived by syntax *partially describes* the form of a fully specific logical form, which in turn represents a context-specific interpretation which can be evaluated against a model or the actual situation at hand.

To map the form of the gesture to an underspecified meaning representation, we use the underspecification formalism of Robust Minimal Recursion Semantics (RMRS, Copestake 2007) – a factorised version of ERG's semantic framework, Minimal Recursion Semantics (MRS, Copestake *et al.* 2005). RMRS was originally developed to support the integration of deep and shallow processing. Modelling gesture is somewhat akin to shallow processing in that one has to handle the large degree of underspecificity.

To illustrate it, consider the MRS for "every dog chased some cat" in (15). Here, the semantic scope ambiguities are captured by the so called *qeq* ($=_q$) contraints which allow for two alternative fully scoped formulas.

(15) $l_1 : every(x_0, h_3, h_1)$
$l_{11} : dog(x_1)$
$l_2 : some(y_0, h_4, h_2)$
$l_{21} : cat(y_1)$
$l_3 : chase(e_1, x_2, y_3), \qquad h_3 =_q l_{11},\ h_4 =_q l_{21}$

While MRS underspecifies scope, it still requires a fully specified predicate-argument structure. However, neither shallow language processors nor gestural form on their own can fully determine a unique predicate argument structure. Refining MRS to RMRS solves this. One simply produces a highly factorised representation of each elementary predication: each one is equipped with its own unique *anchor (a)*, which serves as a locus for specifying the predicate's arguments; equations (e.g., $x_0 = x_1 = x_2$) are also added to express unifiability between variables. So (16) is a notational variant of (15).

(16) $l_1 : a_1 : every(x_0), l_1 : a_1 : RSTR(h_3), l_1 : a_1 : BODY(h_1)$
$l_{11} : a_{11} : dog(x_1)$
$l_2 : a_2 : some(y_0), l_2 : a_2 : RSTR(h_4), l_2 : a_2 : BODY(h_2)$
$l_{21} : a_{21} : cat(y_1)$
$l_3 : a_3 : chase(e_1), l_3 : a_3 : ARG1(x_2), l_3 : a_3 : ARG2(y_3)$
$h_3 =_q l_{11},\ h_4 =_q l_{21}$
$x_0 = x_1 = x_2, \qquad y_0 = y_1 = y_3$

For instance, a POS tagger would yield (17) instead of the more specific (16). Proposition (17) captures the semantic insight that, for example, knowing that the word *chase* is tagged as a verb, one knows that its semantic index is an event, but one does not know how many arguments the predicate symbol introduced by *chase* takes because the POS tagger lacks information about lexical subcategorisation.

(17) $l_1 : a_1 : every(x_0)$
$l_{11} : a_{11} : dog(x_1)$
$l_2 : a_2 : some(y_0)$
$l_{21} : a_{21} : cat(y_1)$
$l_3 : a_3 : chase(e_1)$

Semantic composition with RMRS follows the semantic algebra of Copestake *et al.* (2001): the predications and *qeq* on the mother are accumulated from those in the daughters and the semantic head daughter has its 'hook' (roughly equivalent to a λ-term) replaced by the semantic index of the non-head.

5.3 *Form-meaning mapping*

5.3.1 Depicting gestures

Following Lascarides and Stone (2009a), mapping the form of a depicting gesture to its meaning involves mapping each feature value pair in the TFS representing its form to an RMRS-based underspecified predication: the ULF of the gesture from Figure 3 is shown in (18).

(18) $l_0 : a_0 : [\mathscr{G}](h)$
$l_1 : a_1 : hand\_shape\_bent(i_1)$
$l_2 : a_2 : palm\_orient\_towards\_down(i_2)$
$l_3 : a_3 : finger\_orient\_towards\_down(i_3)$
$l_4 : a_4 : hand\_location\_lower\_periphery(i_4)$
$l_5 : a_5 : hand\_movement\_circular(i_5)$
$h =_q l_n$ *where* $1 \leq n \leq 5$

Each predicate has a label, an anchor, and a semantic index, as is standard in RMRS. Since a predication mapped from depicting gesture could resolve in context to an event e or an individual x, its semantic index is a metavariable i that generalises over e or x. The predicate symbols underspecify the particular constructor and its arity in the LF. For instance, a feature-value pair like $\left[\text{HAND-MOVEMENT}\ \ \text{circular}\right]$ would

map to $l_1 : a_1 : \mathit{hand\_movement\_circular}(i)$. Resolving these predicates happens outside the grammar as a byproduct of discourse processing (Lascarides and Stone 2009a). In particular, each underspecified predicate (such as $\mathit{hand\_movement\_circular}(i)$) is a root to a type hierarchy of increasingly specific predications of content. This is roughly analogous to constructing a specific lexical meaning out of a polysemous lexical entry (Copestake and Briscoe 1995), but here the type hierarchy captures constraints on interpretation that are imposed by the requirement for iconicity – i.e., a resemblance between the form of the gesture and its meaning. This type hierarchy is designed so that a circular hand movement can never resolve to, say, a rectangular concept. To illustrate the idea, in Section 2.2 we claimed that one of the interpretations of the circular hand movement in (1) was the mud being mixed. This is achieved by resolving $\mathit{hand\_movement\_circular}(i)$ to a conjunction of predications: $\mathit{substance}(x') \wedge \mathit{rotate}(e', x')$, which is a node in the type hierarchy that's rooted at $\mathit{hand\_movement\_circular}(i)$, and is featured in (5). In an alternative interpretation this hand movement is a depiction of the mixing event from the agent's viewpoint: i.e., the underspecified predicate $\mathit{hand\_movement\_circular}(i)$ can resolve to the three-place predicate $\mathit{rotate}(e', j', x')$, featured in (7).

Further, recall from Section 2.1 the constraint that an individual that is introduced in a depicting gesture can't be an antecedent to a pronoun in speech. Lascarides and Stone (2009a) regiment this constraint by introducing the scopal operator $[\mathscr{G}]$: all predicates mapped from depicting gesture fall within its scope (via the scopal condition $h =_q l_n$), and the dynamic semantics Lascarides and Stone assign to $[\mathscr{G}]$ ensures that co-reference across the modalities is suitably constrained.

5.3.2 Deictic gestures

The mapping of deixis form to a ULF captures the fact that deixis provides the spatial reference of an individual or event in the physical space $\vec{p}$ (the complete RMRS logical form mapped from the gesture in Figure 4 is shown in (19)). This is formalised by the two-place predicate $l_{21} : a_2 : \mathit{sp\_ref}(i_1)\ \ l_{21} : a_2 : \mathit{ARG1}(v(\vec{p}))$ whose first argument is the underspecified variable i_1, and the second argument ARG1 – linked through the anchor a_2 – is the actually denoted space $v(\vec{p})$ with v being the function that maps the gesture space to the space in denotation (recall discussion in Section 3.2). The ULF is only a partial description

of the resolved LF: e.g., resolving the underspecified referent i_1 to an object x and inferring a relation between the deixis denotation and the speech denotation is a matter of pragmatic reasoning. Note how in the prior interpretation of *hand_movement_ciricular*(i), i resolves to an individual x, whereas here it resolves to an event e.

To capture how the form of the pointing hand affects its meaning, we map each deixis feature-value pair to a two-place predicate, with the first argument being an event variable ($e_0 ... e_n$) and the second argument ARG1 being the referent identified by the pointing signal ($i_0 ... i_n$). This formalisation is similar to the treatment of non-scopal modification in the English Resource Grammar (ERG, Flickinger 2000): a deictic predication (as mapped from form) is a two-place predication whose second argument ARG1 is equated with the semantic index of the modified predication, obtained by equating $i_0 = i_1 = i_2 = i_3 = i_4 = i_5 = i_6$ and whose label is equated with the label of the modified predication, obtained via $l_{21} = l_{22} = l_{23} = l_{24} = l_{25} = l_{26}$. For consistency with ERG where individuals are all bound by quantifiers, we use the *deictic_q* quantifier to quantify over the spatial referent i_1.

(19) $l_1 : a_1 : \mathit{deictic\_q}(i_0)\ l_1 : a_1 : \mathit{RSTR}(h_1)\ l_1 : a_1 : \mathit{BODY}(h_2)$
$l_{21} : a_2 : \mathit{sp\_ref}(i_1)\ l_{21} : a_2 : \mathit{ARG1}(v(\vec{p}))$
$l_{22} : a_3 : \mathit{hand\_shape\_flat}(e_0)\ l_{22} : a_3 : \mathit{ARG1}(i_2)$
$l_{23} : a_4 : \mathit{palm\_orient\_towards\_centre}(e_1)\ l_{23} : a_4 : \mathit{ARG1}(i_3)$
$l_{24} : a_5 : \mathit{finger\_orient\_away\_centre}(e_2)\ l_{24} : a_5 : \mathit{ARG1}(i_4)$
$l_{25} : a_6 : \mathit{hand\_movement\_down}(e_3)\ l_{25} : a_6 : \mathit{ARG1}(i_5)$
$l_{26} : a_7 : \mathit{hand\_location\_c}(e_4)\ l_{26} : a_7 : \mathit{ARG1}(i_6)$
$h_1 =_q l_{21}$
$l_{21} = l_{22} = l_{23} = l_{24} = l_{25} = l_{26}$
$i_0 = i_1 = i_2 = i_3 = i_4 = i_5 = i_6$

6 GRAMMAR RULES FOR SPEECH AND GESTURE

In this section, we propose grammar construction rules that integrate the form of the gesture and the form of the speech signal into a single syntax tree that in turn provides the basis for deriving a ULF of the speech-gesture action. The construction rules license particular speech-gesture alignments, and constraints on their application make

predictions about well-formedness, as motivated via the qualitative observations about speech-gesture data in Section 3.

6.1 *Prosodic word and gesture alignment*

We begin with the straightforward case where gesture aligns with a single lexical item:

Construction Rule 1 (Situated Prosodic Word Constraint)**.** *A depicting or deictic gesture can attach to a spoken word* w *of a spoken utterance if (a.) there is an overlap between the temporal performance of the gesture stroke and* w*; and (b.)* w *bears a nuclear or a pre-nuclear pitch accent.*

We represent the mulitmodal rules as phrase structure rules equipped with the following information (Figure 5): the speech daughter S-DTR and the gesture daughter G-DTR each introduce a TIME feature, a SYNSEM|CAT feature which captures its syntacic category (note that for gestures, this information includes the form feature-value pairs, discussed in Section 5.1) and a SYNSEM|CONT feature

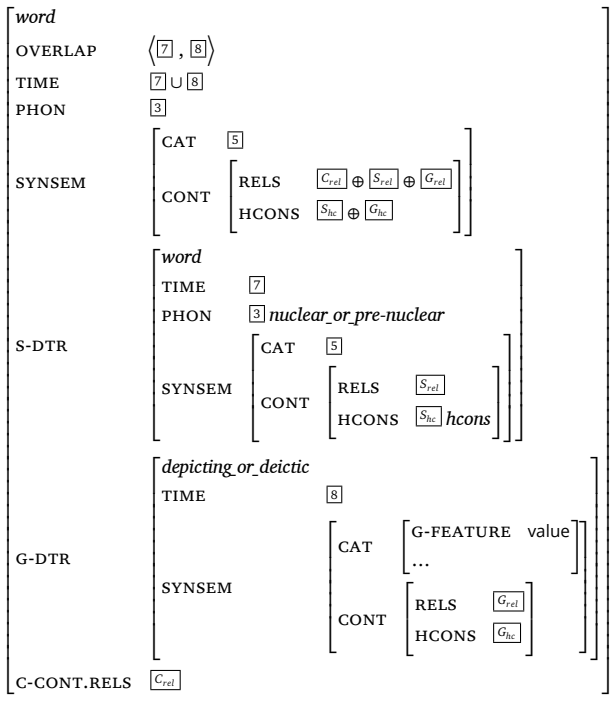


Figure 5:
HPSG-based formalisation of the Situated Prosodic Word Constraint aligning gesture and a spoken word

which captures its (underspecified) semantic contribution. The speech daughter also introduces a PHON feature which captures the phonological information. The construction rule introduces a feature OVERLAP whose values are re-entrant with values in the temporal components of the daughters; and also a TIME feature which is the union of the speech daughter's value and the gesture daughter's value. In so doing, we follow previous work where timing is used as a constraint on the integration (Johnston *et al.* 1997). As it is standardly done in ERG, the semantic contribution of the construction rule is captured within C-CONT: here, a depicting gesture introduces an underspecified relation *vis_rel* between the main label of the gesture semantics and the main label of the semantics of the spoken phrase; the underspecified relation introduced by deixis is *deictic_rel* between the semantic index of the speech daughter and the semantic index of the gesture daughter. Multimodal integration happens via unification of these features.

Given the different form-meaning mappings of depicting vs. deictic gestures, we will now provide separate analyses for both gesture types.

6.1.1 Situated Prosodic Word Constraint and depicting gesture

To illustrate how the Situated Prosodic Word Constraint works with depicting gestures, consider again example (1). The nuclear accent is on the rightmost word "mud", which licenses an attachment of the gesture to it using Construction Rule 1. The derivation, which attaches the gesture to "mud", is shown in Figure 6.

The prosodic PHON and syntactic CAT information of the speech head daughter gets propagated to the mother node. We do not propagate the gesture form features to the mother node since we do not need to access gesture form any further. The timing of the situated utterance is recorded in the mother's TIME value. This information is necessary in case the (situated) word aligns with another gesture.

The semantic composition follows the standard English Resource Grammar (ERG) process, namely: the individual semantic formulae are decorated with a global label (h_1) which demonstrates the derivation of a single LF. Each formula is also augmented with a hook containing the local top label (LTOP, equated to the label of the main predication) and the semantic index. The LTOP of the predicate contributed by the speech daughter $l_6 : a_6 : \_mud\_n\_1(x_1)$ is l_6 and the index is x_1. The

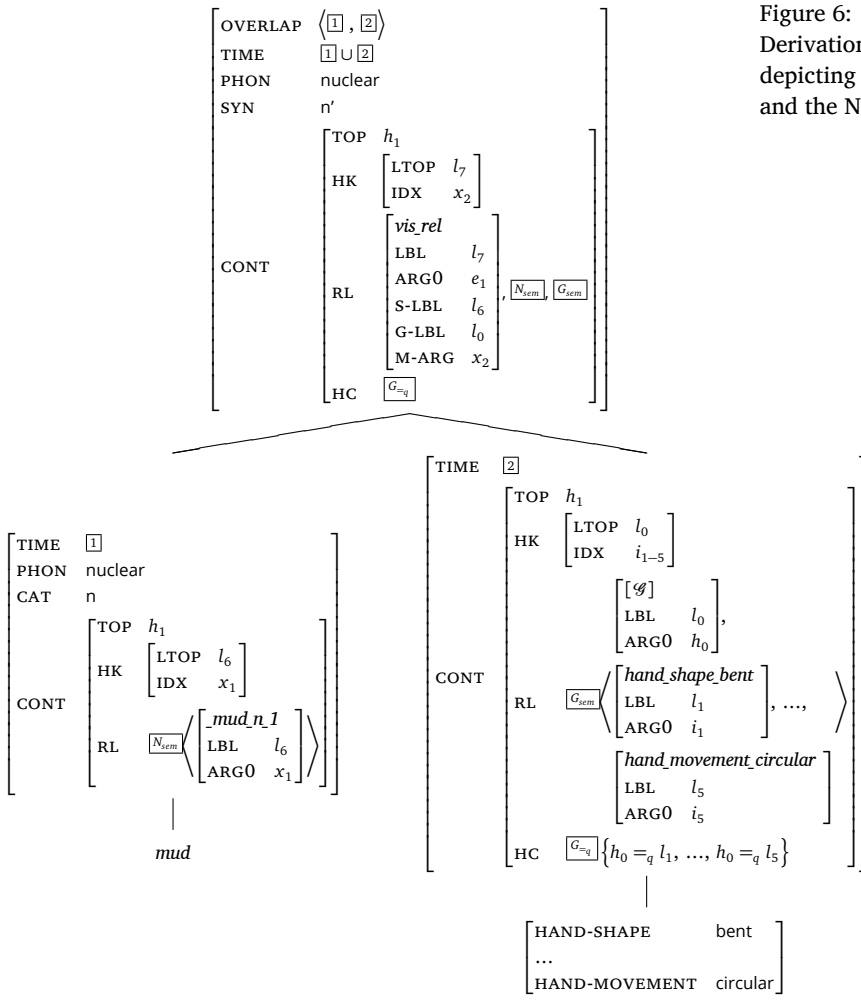


Figure 6: Derivation tree for depicting gesture and the N “mud”

LTOP of the gesture daughter is equated to the label of the $\mathscr{G}$ modality – l_0. Regarding the gesture semantic index, the gesture LF is too underspecified to know which of the semantic predications will resolve to the main variable and hence at this stage we have no information as to which is the semantic index of the formula. We therefore use i_{1-5} as a shorter notation for a disjunction of co-indexations to reflect the fact that the underspecified variable $i_1 \ldots i_5$ of each gesture predicate could potentially resolve to the main variable: event e or individual x.

Note that the semantic representation CONT of the situated ut-

terance which features the underspecified relation *vis_rel* between the top label l_6 of the speech daughter and the top label l_0 of the gesture daughter to designate that the speech and gesture are coherently connected. In RMRS, labels denote the scopal position of an elementary predication. We therefore code the arguments of *vis_rel* as S-LBL and G-LBL to designate that their values are labels of spoken and gestural predications, respectively. As illustrated in Section 2.1, *vis_rel* is resolvable at the semantics/pragmatics interface to a specific value – e.g., Depiction, Elaboration – that is dependent on resolving the gestural denotation. Here, the attachment to "mud" would support an interpretation where the gesture designates some substance and the fact that it was going round, which in turn would resolve *vis_rel* to Elaboration, as featured in the LF in (5). The truth conditional contribution of the gesture will thus ultimately be roughly analogous to an appositive or a non-restrictive relative clause modifying the noun. Note that given constraints on reference on the semantics/pragmatics interface, this attachment blocks the gesture referring to anything that is bridging related to "mixes" or "he".

The CONT of the mother is obtained by equating the TOP of the mother to the TOP of the daughters. The relations (abbreviated as RL) of the situated phrase are equal to the append of the predications of the gesture daughter $\boxed{\text{G}_{sem}}$ and the speech daughter $\boxed{\text{N}_{sem}}$, and also *vis_rel*. Further, *vis_rel* introduces a multimodal argument M-ARG which serves as a semantic index of the integrated speech-gesture signal (the hook's index is therefore equated to the index of M-ARG – x_2), and so it can be taken as an argument by any external predicate. Here, for instance, the verb "mix" would take two arguments: ARG1 – corresponding to the subject – would be identified with ARG0 of "he", and ARG2 – corresponding to the object – would be identified with M-ARG of the situated word, consisting of "mud" and the gesture.

6.1.2 Situated Prosodic Word Constraint and deictic gesture

We illustrate the syntactic derivation and the semantic composition for deixis and a spoken word using utterance (14). The derivation tree is shown in Figure 7. The Situated Prosodic Word Constraint licenses an attachment of the deictic gesture to the verb "enter": it is marked by a pre-nuclear accent, and it temporally overlaps the gesture.

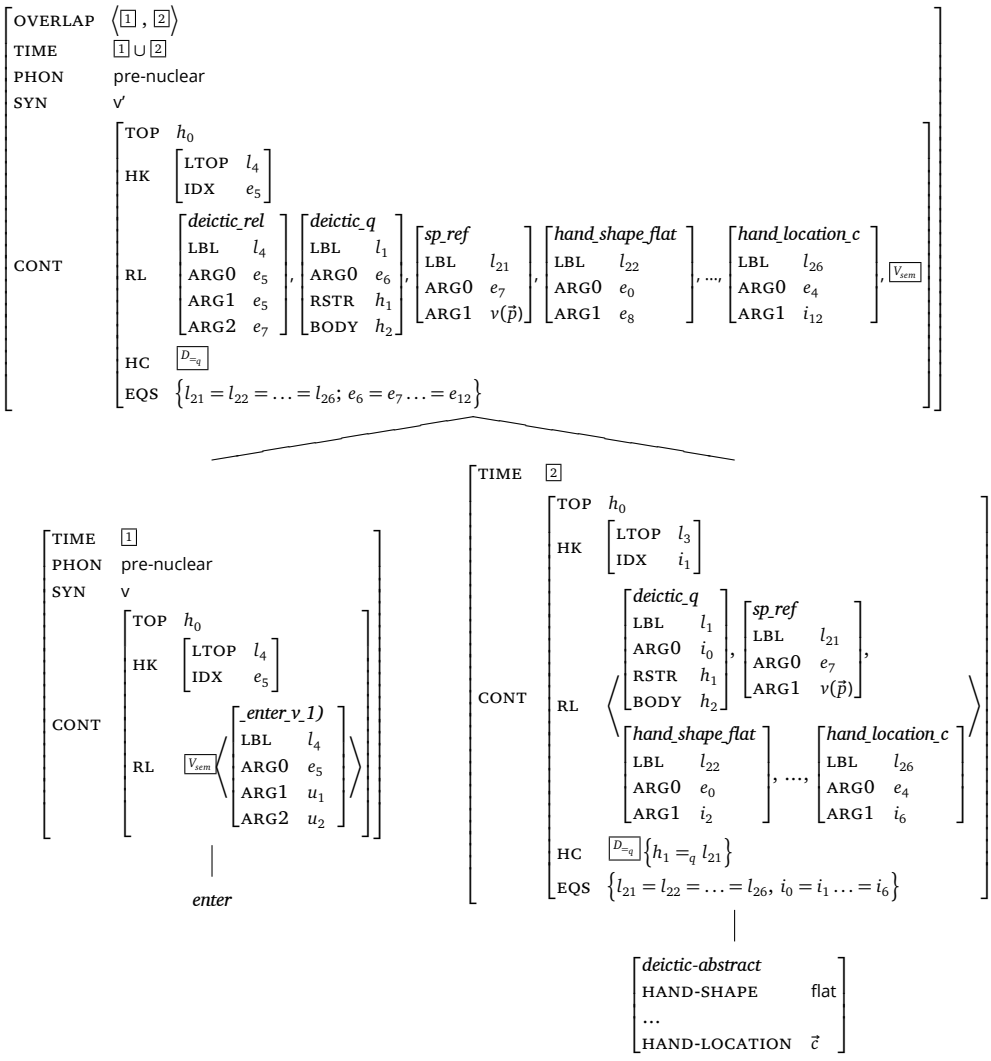


Figure 7: Derivation tree for deictic gesture and the V “enter”

The semantic composition proceeds in the same way as with depicting gestures. Since the gesture semantics features a quantifier (*deictic_q*), the local top of gesture is distinct from the label of the quantifier. The semantic index is the underspecified variable i_1 bound by *sp_ref*. In composition, the deixis semantic predicates (as shown

in 19) append to the semantic predicate $\boxed{\mathrm{V}_{sem}}$ of the speech daughter – $l_4 : a_9 : \_enter\_v\_1(e_5)$ $l_4 : a_9 : ARG1(u_1)$ $l_4 : a_9 : ARG2(u_2)$. In so doing, the underspecified semantic index i_1 of the deixis unifies with the semantic index e_5 of the speech, and so the underspecified gesture variable i_1 of $sp\_ref(i_1)$ resolves to an event (e_7).

Like depicting gestures, deictic gestures are connected in semantics to their aligned speech via an (underspecified) relation. The construction rule therefore introduces the underspecified relation $deictic\_rel(e_5, e_7)$ between the semantic index e_5 of the speech predication and the semantic index e_7 of the deictic gesture. Pragmatics must then resolve this relation to a specific value: one possible resolution would be VirtualCounterpart – i.e., the deictic gesture denotes a virtual counterpart of the coordinates of entering the apartment door. Similarly to the treatment of non-scopal modification in language, this relation shares the same label as the speech head daughter since it further restricts the referent introduced by the gesture. Informally, the gesture here functions as an appositive in language and a rough linguistic paraphrase is "the entering event, the event at the coordinates pointed at".

6.2 *Speech phrase and gesture alignment*

One of our central claims is that ambiguities as to which speech phrase a co-speech gesture aligns with are best modelled as attachment ambiguities within the grammar. As we demonstrated in Section 2.2, the relative timing of speech and gesture is not the only constraint on using such construction rules; also, temporal constraints should be weaker than *simultaneity*, contrary to McNeill (1992). Rather, we argued that the gesture should temporally overlap with its aligned speech (if it didn't, then by definition it wouldn't be co-speech gesture!) and furthermore temporally overlap with an *accented element* in the (aligned) speech unit. Thus a single utterance such as (1) or (14) can licence different speech-gesture alignments, each of them supporting a distinct range of plausible pragmatic interpretations in accordance with constraints on reference (see Section 2.1). Likewise, it is perfectly acceptable for the gesture in (1) to be performed only while uttering the accented word "mud", and still interpret the gesture in all the ways proposed in Section 2.2. In this section we provide the formal methodology of how to arrive at these interpretations.

As proposed in Section 2.2, we introduce construction rules that allow a gesture to align with an *entire constituent* – that is, a head combined with its arguments – in contrast to Rule 1 that aligns gesture with a (temporally overlapping, accented) word. From a descriptive perspective, the inclusion of more context into the speech aligned with gesture is grounded in the "synthetic" nature of gesture versus the "analytic" nature of the spoken words (McNeill 2005). For instance, in example (1) the information about the direction of the mixing event (i.e., clockwise, downwards), the manner of performing the mixing action (i.e., using the entire hand) is denoted by a single visual performance and by several linearly ordered lexical items ("mixes", "mud"). For the purposes of a multimodal grammar it is essential to distinguish between temporal synchrony and alignment: whereas the former is a quantitative measurement of when the two modalities happen, the latter is a qualitative, linguistic notion pertaining to the syntax tree of speech and gesture and the meaning representation it corresponds to. By setting apart these two notions, we also ensure that the physical termination of the gesture does not enable attachment to a midpoint of a speech constituent.

With all this in mind, we now define the construction rule that allows a gesture to attach to a constituent larger than a single prosodic word:

Construction Rule 2 (Situated Spoken Phrase Constraint)**.** *A depicting or deictic gesture can attach to any of the higher projections in the derivation tree of the nuclear/pre-nuclear accent element, which also form a syntactic and/or prosodic constituent xp, no matter what the syntactic label is if there is an overlap between the temporal performance of the gesture stroke and xp.*

The attachment of the gesture to any projection in the tree would allow for saturating the head with its selected arguments before the attachment takes place. This means that the attachments are licensed at each saturation step. In this way, we account for the fact that gesture can co-refer to any or all of these arguments in the fully resolved pragmatic interpretation. Note also that Rule 2 used 'syntactic and/or prosodic constituent' to refer to any phrase of a hierarchical organisation: prosodic or syntactic. Assuming an analysis where there is no isomorphism between syntax and prosody, this flexibility is necessary

whenever there are mismatches between prosodic structure and syntactic structure.[8]

Since the attachments of depicting gesture to a speech phrase are analogous to the attachments of deixis to the speech phrase, we illustrate the possible attachments using the depicting gesture in utterance (1). Recall from Section 2.2 that the resolved LFs for this speech-gesture action featured coherence relations between: (i) the NP's denotation and the 'rotating' gesture, and (ii) between the VP's (or S's) denotation and the 'rotating' gesture. We discussed (i) in the previous section and we therefore forego any further details about it. Given the construction rule in 2, interpretation (ii) is supported as follows: attach the gesture to VP "mixes mud" (or to the S "he mixes mud"). In both cases, the gesture stroke temporally overlaps the nuclear prominent "mud", and so the gesture can attach to its VP projection or S projection. Both of these attachments force the gesture to qualify "mixes" (for the second argument to the underspecified coherence relation that's introduced by the construction rule must outscope $mix(e, y, x)$). They underspecify, however, the relative scope of the coherence relation with respect to the predication $mud(x)$ and $pron(y)$. If these resolve to being within the scope of the coherence relation, then the resolved interpretation of the gesture can co-refer to *he* and to the mud; if not, it can't.

Further to this, we claimed that utterance (10) was ill-formed since the gesture was performed along a non-accented item in an all-rheme utterance. Having introduced the construction rules 1 and 2, we are now in a position to account for the utterance's ill-formedness: the form of (10) doesn't meet the constraints for either of our construction rules. On the other hand, if the gesture was performed in a way that temporally overlaps the prosodic word "mother", then the rules we've proposed license attachments to the N "mother", the NP "your mother" and even to the S "your mother called".

[8] In prior work on HPSG-based analysis of prosody (Klein 2000), prosodic structures are analysed in parallel with syntactic structures.

6.3 *Spoken word and gesture alignment: temporal and prosodic relaxation*

The two construction rules we've proposed allow a co-speech gesture to align with a prosodic word or with a constituent that contains prosodic element(s) that overlap the temporal performance of the gesture. These constructions, however, are not sufficient as they do not reflect an important finding from our data. We used examples (12) and (13) to illustrate that when the referent of the deictic gesture is visually salient, the deictic gestures does *not* necessarily overlap a prosodically prominent word and/or temporally overlap the semantically related word. The following rule takes this into account.

Construction Rule 3 (Deictic Prosodic Word with Defeasible Constraint)**.** *The constraints on temporal overlap in 1 and 2 are defeasible, i.e., a deictic gesture attaches to a word that is not prosodically prominent and/or whose temporal performance is adjacent to that of the deictic stroke if: (a.) the mapping v from gestured space $\vec{p}$ to space in denotation $v(\vec{p})$ resolves to equality; and (b.) the temporal performance of the gesture overlaps (some portion of) the spoken utterance containing the word.*

This temporal/prosodic relaxation rule integrates a defeasible constraint with the view of producing LFs that in context resolve to the intended meaning. As attested by (13),[9] the relaxation of this contraint depends on the salience of co-present individuals and it is thus necessary only in utterances where the gesture denotation is physically present in the visible space, i.e., there is an equality between the physical space that the hand points at and the gesture referent. This rule accounts for the fact that certain characteristics of the context (i.e., salience of the individual pointed at) are required for the rule to apply. Otherwise, the interpretation could be infelicitous. Similar issues occur with deictic expressions and other referential expressions which require a salient individual in context for the utterance to be felicitous (see Lücking *et al.* 2006a).

Note also that this rule constrains the alignment to temporal overlap between (some portion of) the utterance and the gesture. This means that the grammar does not handle gestures performed either before or after the temporal performance of the utterance since any-

[9] Many more examples can be found in the AMI corpus.

thing beyond the clausal level is a matter of relating discourse units. For instance, while the temporal overlap between the gesture and the speech signal in (13) takes care of aligning the gesture and the semantically related element – i.e., "she" in (13) – the gesture in (12) does not overlap any portion of the utterance containing "mouse" and hence the grammar rule cannot attach the gesture to the noun "mouse". Similarly to relating purely linguistic discourse segments, relating the gesture in (12) with the noun "mouse" is a matter of discourse processing that lies beyond the scope of the (syntactic) grammar.

With this constraint in mind, let us examine the possible derivations of utterance (13). The Situated Prosodic Word 1 would license attachments to the temporally overlapping prosodically prominent "said". Although syntactically well-formed, this attachment would not produce the contextually preferred (and the most intuitive) interpretation: namely, an identity between the gesture referent and the speech referent. An alternative attachment is provided by Construction Rule 3: the deictic gesture may attach to "she" thereby providing an interpretation where the gesture denotation is identical to the denotation of the pronoun "she".

7 IMPLEMENTATION AND EVALUATION

The main challenge for the grammar implementation stems from the non-linear input of speech-and-gesture actions. Existing grammar engineering platforms for unification-based grammars typically only parse linearly ordered strings, and so they do not handle multimodal signals whose input comes from separate channels connected through temporal relations. Also, these parsing platforms do not support quantitative comparison operations over the time stamps of the input tokens. This is essential for our grammar since temporal overlap constraints choices of attachment.

To solve this, we pre-processed the XML-based Feature Structure (FS) input so that overlapping TIME values were 'translated' into identical start and end edges of the speech token and the gesture token as follows:

```
<edge source="v0" target="v1">
        <fs type="speech_token">
<edge source="v0" target="v1">
        <fs type="gesture_token">
```

This pre-processing step is sufficient since the only temporal relation required by the grammar is *overlap*, an abstraction over more fined-grained relations between speech (S) and gesture (G) such as (*precedence(start(S), start(G)) ∧ identity (end(S), end(G))*).

The linking of gesture to its temporally overlapping speech segment happens prior to parsing via chart-mapping rules (Adolphs *et al.* 2008) which involve re-writing chart items into FSs. The `gesture-unary-rule` (Figure 8) rewrites an input (I) speech token in the context (C) of a gesture token into a combined speech + gesture token where the +GEST and +PROS values of the speech and gesture tokens are copied onto the output (O).

```
gesture-unary-rule := cm_rule &
 [+CONTEXT <gesture_token & [+GEST #gest]>,
  +INPUT <speech_token & [+PROS #pros]>,
  +OUTPUT <speech+gesture_token &
        [+GEST #gest, +PROS #pros]>,
  +POSITION "O1@I1, I1@C1" ].
```

Figure 8:
Definition of `gesture-unary-rule`

The +PROS attribute contains prosodic information and the +GEST attribute is a feature-structure representation. The +POSITION constraint restricts the position of the I, O and C items to an overlap (@), i.e., the edge markers of the gesture token should be identical to those of the speech token, and also identical to the speech + gesture token. This chart-mapping rule recognises the gesture token overlapping the speech token and it records this by "augmenting" the speech token with the gesture feature-values.

Gestures overlapping more than one speech token were handled by further chart-mapping rules that distributed the gestural information onto multiple speech tokens within the temporal span of the gesture. So a gesture overlapping, say, three speech tokens, would get split into three gesture tokens. Then, the `gesture-unary-rule` was applied so as to instantiate a speech + gesture token for each speech token temporally overlapping the gesture. The result of this chart-mapping operation is multiple gesture-marked speech tokens whose span is identical to the span of the gesture.

A separate rule was also required for concrete deixis to account for the permitted precedence and sequence relations between the speech token and the concrete deictic gesture token. This rule (which we omit

for the sake of space) remains neutral about the positional (and hence temporal) relation between the gesture token and the speech token, thus allowing a gesture token of type *deictic-concrete* to attach to each speech token from the input chart.

In the grammar, we extended the ERG word and phrase rules with prosodic and gestural information where the +PROS and +GEST features of the input token are identified with the PROS and GEST of the word and/or lexical phrase in the grammar. We then added a gesture lexical rule (Figure 9) which projects a gesture daughter to a complex gesture-marked entity for which both the PROS and GEST features are appropriate.

Figure 9: Definition of `gesture_lexrule`

```
gesture_lexrule := phrase_or_lexrule &
 [ ORTH [ PROS #pros,
          GEST no-gesture],
   ARGS <[ ORTH [ GEST gesture-form,
                  PROS p-word & #pros ]]>].
```

In line with Definition 1, this rule constrains PROS to a prosodically prominent word of type *p-word* thereby preventing a gesture from plugging into a prosodically unmarked word. The *gesture-form* value is a supertype over the distinct gesture types – depicting and deictic. The GEST feature of the mother is of type *no-gesture* to block any further recursive instantiation of this rule. The `gesture_lexrule` is inherited by a lexical rule specific to depicting gestures, and by a lexical rule specific to deictic gestures. In this way, we can encode the semantic contribution of depicting gestures which is different from the semantic contribution of deixis. For the sake of space, Figure 10 presents only the `depicting_lexrule`. The semantic information contributed by the rule is encoded within C-CONT.

The rule introduces an underspecified *vis_rel* between the main label `#dltop` of the spoken sign (via the HCONS constraints) and the main label `#glbl` of the gesture semantics (via the HCONS constraints). Note that these two arguments are in a *geq* (greater or equal) constraint. This means that *vis_rel* can operate over any projection of the speech word; e.g., attaching the gesture to "mud" in (1) means that the relation is not restricted to the EPs contributed by "mud" but it can be also be over the EPs of a higher projection. Here, the implemented analysis differs from the theoretical one in that we formalise

```
depicting_lexrule := gesture_lexrule &
[ARGS <[ SYNSEM.LOCAL.CONT.HOOK.LTOP #dltop,
          ORTH [ GEST depicting] >,
 C-CONT [ RELS <![ PRED vis_rel,
                   S-ARG #arg1,
                   G-ARG #arg2 ],
                 [ PRED G_mod,
                   LBL #glbl,
                   ARG1 #harg ],
                 [ LBL #larg1 ],...!>,
          HCONS <!geq&[ HARG #arg1,
                        LARG #dltop ],
                  qeq&[ HARG #arg2,
                        LARG #glbl ],
                  qeq&[ HARG #harg,
                        LARG #larg1 ],
                  ...!>]].
```

Figure 10:
Definition of `depicting_lexrule`

in semantics the gesture attachment ambiguities as per Situated Spoken Phrase Constraint: that is, *vis_rel* can operate over any projection of the gesture-marked sign.

The gesture's semantics is a bag of EPs, all of which are outscoped by the gestural modality [$\mathscr{G}$]. The rule therefore introduces in RELS a label (here `#larg1`) for an EP which is in *qeq* constraints with [$\mathscr{G}$]. The instantiation of the particular EPs comes from the gestural lexical entry. In the real implementation, the number of these labels corresponds to the number of features.

The evaluation was performed in the tradition of testing wide-coverage grammars, by means of a manually crafted test suite (Oepen *et al.* 1997). We created a test suite covering different gesture types, prosody and the following linguistic phenomena: intransitivity, transitivity, complex NPs, modification, negation and coordination. The test set contained 471 speech-gesture items (71.5% well-formed; 28.5% ill-formed) covering the full range of prosodic (prosodic markedness and unmarkedness) and gesture (the span of depicting/deictic gesture and its temporal relation to the prosodically marked elements) permutations. The gestural vocabulary was limited since a larger gesture lexicon has no effects on the performance. To test the grammar, we used the [incr tsdb()] competence and performance tool (Oepen 2001) which enables batch processing of test items and which creates a cov-

Table 1: Gesture grammar coverage profile of test items generated by [incr tsdb()]

Aggregate	total items ♯	positive items ♯	word string ϕ	lexical items ϕ	distinct analyses ϕ	total results ♯	overall coverage %
$90 \leq$ *i-length* < 95	126	91	93.00	26.41	1.89	91	100.0
$70 \leq$ *i-length* < 75	78	53	71.00	12.00	1.00	53	100.0
$60 \leq$ *i-length* < 65	249	179	60.00	9.42	1.00	179	100.0
$45 \leq$ *i-length* < 50	18	14	49.00	7.00	1.00	14	100.0
Total	471	337	70.18	14.31	1.24	337	100.0

erage profile of the test set (see Table 1). The values are as follows: the left column separates the items per aggregation criterion (the length of test items);[10] the next column shows the number of test items per aggregate; then we have the number of grammatical items; average length of test item; average number of lexical items; average number of distinct analyses and total coverage.

We manually verified the coverage. While the grammar successfully parses all well-formed examples, the inclusion of a separate chart-mapping rule for concrete deixis results in overgeneration. We believe that the alternative method of enforcing strict precedence or strict sequence is too restrictive with respect to the possible interpretations supported by the distinct attachment configurations.

Finally, we also verified that the newly introduced rules did not change the coverage or increase the ambiguity of the existing broad-coverage grammar. We therefore ran both the ERG grammar and the gesture grammar on the ERG testsuite. The results shown in Table 2 were generated by both the ERG grammar and by the grammar equipped with the gesture rules. In other words, the gesture rules had no effects on the existing rules.

8 CONCLUSIONS

The work presented here advances a new theory in which the form-meaning mapping of speech-gesture actions was analysed using well-established methods from linguistics such as constraint-based syntactic derivation and semantic composition. In particular, we cap-

[10] Note the length here does not correspond to the actual length of tokens in each test item, since the tool also counts the XML tags.

Aggregate	total items ♯	positive items ♯	word string ϕ	lexical items ϕ	distinct analyses ϕ	total results ♯	overall coverage %
$55 \leq$ *i-length* < 60	3	3	55.00	108.00	2.00	3	100.0
$45 \leq$ *i-length* < 50	7	7	49.00	69.00	16.86	7	100.0
$40 \leq$ *i-length* < 45	17	17	43.00	69.50	4.94	16	94.1
$35 \leq$ *i-length* < 40	32	32	37.00	41.87	2.84	32	100.0
$30 \leq$ *i-length* < 35	30	30	31.00	32.57	2.37	30	100.0
$25 \leq$ *i-length* < 30	13	13	25.00	42.00	1.67	12	92.3
$15 \leq$ *i-length* < 20	13	13	19.00	15.58	1.83	12	92.3
Total	115	115	34.13	43.99	3.63	112	97.4

(generated by [incr tsdb()] at 8-jul-2005 (04:42 h))

Table 2: [incr tsdb()] coverage profile of ERG test items parsed by ERG and gesture grammar

tured the mapping of form of speech-gesture actions to their meanings within a constraint-based grammar: the construction rules were inspired by examining real data and were further implemented within a wide-coverage grammar for English. The highly ambiguous gesture form was captured using underspecified semantics, which allowed us to account for the range of specific interpretations that a given gesture can take in its context of use. The ambiguities notwithstanding, we demonstrated that the speech-gesture attachments are constrained by the form of the speech signal, thus showing that the difference in ambiguity between linguistic input and gesture input is more a matter of degree than a difference in kind.

ACKNOWLEDGEMENTS

The authors are very grateful to Elżbieta Hajnicz and the anonymous reviewers, Daniel Loehr, Matthew Stone, Mark Steedman, Emily Bender, Bob Ladd, Michael Johnston, Jonathan Kilgour, Ulrich Schäfer, Stephan Oepen, and also EPSRC for funding this work, as well as ERC (grant number 269427).

REFERENCES

Dorit ABUSCH (2014), Temporal Succession and Aspectual Type in Visual Narrative, in Luka CRNIČ and Uli SAUERLAND, editors, *The Art and Craft of Semantics: A Festschrift for Irene Heim*, volume 1, pp. 9–29, MIT Working Papers in Linguistics, Cambride, MA.

Peter ADOLPHS, Stephan OEPEN, Ulrich CALLMEIER, Berthold CRYSMANN, Daniel FLICKINGER, and Bernd KIEFER (2008), Some Fine Points of Hybrid Natural Language Parsing, in *Proceedings of the Sixth International Language Resources and Evaluation*, ELRA.

Stergos AFANTENOS, Eric KOW, Nicholas ASHER, and Jeremy PERRET (2015), Discourse parsing for multi-party chat dialogues, in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics*, pp. 928–937, Lisbon.

Hiyan ALSHAWI (1992), *The Core Language Engine*, Cambridge: MIT Press.

Nicholas ASHER and Alex LASCARIDES (1998), Bridging, *Journal of Semantics*, 15(1):83–113.

Nicholas ASHER and Alex LASCARIDES (2003), *Logics of Conversation*, Cambridge University Press.

Janet Beavin BAVELAS and Nicole CHOVIL (2006), Hand gestures and facial displays as part of language use in face-to-face dialogue, in V. MANUSOV and M. PATTERSON, editors, *Handbook of Nonverbal Communication*, pp. 97–115, Thousand Oaks, CA: Sage.

Johan BOS (2004), Computational Semantics in Discourse: Underspecification, Resolution, and Inference, *J. of Logic, Lang. and Inf.*, 13(2):139–157, ISSN 0925-8531, doi:10.1023/B:JLLI.0000024731.26883.86, `http://dx.doi.org/10.1023/B:JLLI.0000024731.26883.86`.

Jean CARLETTA (2006), Announcing the AMI Meeting Corpus, *The ELRA Newsletter*, 11(1):3–5.

Jean CARLETTA (2007), Unleashing the killer corpus: experiences in creating the multi-everything AMI Meeting Corpus, *Language Resources and Evaluation*, 41(2):181–190.

Justine CASSELL, David MCNEILL, and K.E. MCCULLOUGH (1999), Speech-Gesture Mismatches: Evidence for One Underlying Representation of Linguistic and Non-Linguistic Information, *Pragmatics and Cognition*, 7(1):1–33.

Ann COPESTAKE (2007), Semantic composition with (robust) minimal recursion semantics, in *DeepLP '07: Proceedings of the Workshop on Deep Linguistic Processing*, pp. 73–80, Association for Computational Linguistics, Morristown, NJ, USA.

Ann COPESTAKE and Ted BRISCOE (1995), Semi-Productive Polysemy and Sense Extension, *Journal of Semantics*, 12:15–67.

Ann COPESTAKE, Dan FLICKINGER, Ivan SAG, and Carl POLLARD (2005), Minimal Recursion Semantics: An introduction, *Journal of Research on Language and Computation*, 3(2–3):281–332.

Ann COPESTAKE, Alex LASCARIDES, and Dan FLICKINGER (2001), An Algebra for Semantic Construction in Constraint-based Grammars, in *Proceedings of the*

39th Annual Meeting of the Association for Computational Linguistics (ACL/EACL 2001), pp. 132–139, Toulouse.

Markus EGG, Alexander KOLLER, and Joachim NIEHREN (2001), The Constraint Language for Lambda Structures, *Journal of Logic, Language and Information*, 10:457–485, ISSN 0925-8531, doi:10.1023/A:1017964622902, http://portal.acm.org/citation.cfm?id=595849.596040.

Randi ENGLE (2000), *Toward a Theory of Multimodal Communication: Combining Speech, Gestures, Diagrams and Demonstrations in Structural Explanations*, Stanford University, PhD thesis.

Dan FLICKINGER (2000), On Building a More Efficient Grammar by Exploiting Types, *Natural Language Engineering*, 6 (1) (Special Issue on Efficient Processing with HPSG):15 – 28.

Ellen FRICKE (2008), *Foundations of a Multimodal Grammar for German: Syntactic Structures and Functions (Grundlagen einer multimodalen Grammatik des Deutschen: Syntaktische Strukturen und Funktionen)*, Europa-Universität Viadrina Frankfurt (Oder), Habilitation, Manuskript. Original document in German.

Gianluca GIORGOLO (2012), Integration of Gesture and Verbal Language: A Formal Semantics Approach, in Eleni EFTHIMIOU, Georgios KOUROUPETROGLOU, and Stavroula-Evita FOTINEA, editors, *Gesture and Sign Language in Human-Computer Interaction and Embodied Communication*, volume 7206 of *Lecture Notes in Computer Science*, pp. 216–227, Springer Berlin Heidelberg, ISBN 978-3-642-34181-6, doi:10.1007/978-3-642-34182-3_20, http://dx.doi.org/10.1007/978-3-642-34182-3_20.

Gianluca GIORGOLO and Ash ASUDEH (2011), Multimodal Communication in LFG: Gestures and the Correspondence Architecture , in Miriam BUTT and Tracy Holloway KING, editors, *The Proceedings of the LFG 2011 Conference*, pp. 257–277, Hong Kong, http://cslipublications.stanford.edu/LFG/16/abstracts/lfg11abs-giorgoloasudeh2.html.

Gianluca GIORGOLO and Frans VERSTRATEN (2008), Perception of speech-and-gesture integration, in *Proceedings of the International Conference on Auditory-Visual Speech Processing 2008*, pp. 31–36.

Erving GOFFMAN (1963), *Behavior in Public Places: Notes on the Social Organization of Gatherings*, The Free Press.

Alex GRZANKOWSKI (2015), Pictures Have Propositional Content, *Review of Philosophy and Psychology*, 6(1):151–163, ISSN 1878-5158, doi:10.1007/s13164-014-0217-0, http://dx.doi.org/10.1007/s13164-014-0217-0.

Florian HAHN and Hannes RIESER (2010), Explaining Speech Gesture Alignment in MM Dialogue Using Gesture Typology, in Paweł ŁUPKOWSKI and Matthew PURVER, editors, *Aspects of Semantics and Pragmatics of Dialogue. SemDial 2010, 14th Workshop on the Semantics and Pragmatics of Dialogue*, pp. 99–109, Polish Society for Cognitive Science, Poznań.

Jerry R HOBBS (1985), On the Coherence and Structure of Discourse, Technical report, Stanford University, Center for the Study of Language and Information.

Michael JOHNSTON (1998a), Multimodal Language Processing, in *Proceedings of the International Conference on Spoken Language Processing (ICSLP)*, Sydney, Australia.

Michael JOHNSTON (1998b), Unification-based Multimodal Parsing, in *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 1*, ACL 1998, pp. 624–630, Association for Computational Linguistics, Stroudsburg, PA, USA, doi:http://dx.doi.org/10.3115/980845.980949, `http://dx.doi.org/10.3115/980845.980949`.

Michael JOHNSTON, Philip R. COHEN, David MCGEE, Sharon L. OVIATT, James A. PITTMAN, and Ira SMITH (1997), Unification-Based Multimodal Integration, in Philip R. COHEN and Wolfgang WAHLSTER, editors, *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 281–288, Association for Computational Linguistics, Somerset, New Jersey.

David KAPLAN (1989), Demonstratives, in J. ALMOG, J. PERRY, and H. WETTSTEIN, editors, *Themes from Kaplan*, Oxford.

Andrew KEHLER (2002), *Coherence, Reference, and the Theory of Grammar*, CSLI Publications.

Ruth KEMPSON, Wilfried MEYER-VIOL, and Dov M GABBAY (2000), *Dynamic syntax: The flow of language understanding*, Wiley-Blackwell.

Adam KENDON (1972), Some relationships between body motion and speech, in A. SEIGMAN and B. POPE, editors, *Studies in Dyadic Communication*, pp. 177–216, Pergamon Press, Elmsford, New York.

Adam KENDON (2004), *Gesture. Visible Action as Utterance*, Cambridge University Press, Cambridge.

Ewan KLEIN (2000), A constraint-based approach to English prosodic constituents, in *ACL '00: Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pp. 217–224, Association for Computational Linguistics, Morristown, NJ, USA, doi:http://dx.doi.org/10.3115/1075218.1075246.

Alexander KOLLER, Michaela REGNERI, and Stefan THATER (2008), Regular tree grammars as a formalism for scope underspecification, in *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-08: HLT)*, Columbus, Ohio.

Stefan KOPP, Paul TEPPER, and Justine CASSELL (2004), Towards integrated microplanning of language and iconic gesture for multimodal output, in *ICMI '04: Proceedings of the 6th international conference on Multimodal interfaces*,

pp. 97–104, State College, PA, USA, ACM, New York, NY, USA, ISBN 1-58113-995-0, doi:http://doi.acm.org/10.1145/1027933.1027952.

Stefan KOPP, Paul A. TEPPER, Kimberley FERRIMAN, Kristina STRIEGNITZ, and Justine CASSELL (2007), *Trading Spaces: How Humans and Humanoids Use Speech and Gesture to Give Directions*, pp. 133–160, John Wiley & Sons, Ltd, ISBN 9780470512470, doi:10.1002/9780470512470.ch8, http://dx.doi.org/10.1002/9780470512470.ch8.

Peter KÜHNLEIN, Manja NIMKE, and Jens STEGMANN (2002), Towards an HPSG-based Formalism for the Integration of Speech and Co-Verbal Pointing, in *Proceedings of Gesture – The Living Medium*, Austin, Texas.

Alex LASCARIDES and Matthew STONE (2006), Formal Semantics for Iconic Gesture, in *Proceedings of Brandial'06, the 10th International Workshop on the Semantics and Pragmatics of Dialogue (SemDial10)*, pp. 125–132, Universitätsverlag Potsdam, Potsdam, Germany.

Alex LASCARIDES and Matthew STONE (2009a), Discourse Coherence and Gesture Interpretation, *Gesture*, 9(2):147–180.

Alex LASCARIDES and Matthew STONE (2009b), A Formal Semantic Analysis of Gesture, *Journal of Semantics*, 26(4):393–449.

Stephen C. LEVINSON (1983), *Pragmatics*, Cambridge University Press, Cambrdige.

Daniel LOEHR (2004), *Gesture and Intonation*, Georgetown University, Washington DC, doctoral dissertation.

Andy LÜCKING, Hannes RIESER, and Marc STAUDACHER (2006a), Multi-modal Integration for Gesture and Speech, in David SCHLANGEN and Raquel FERNÁNDEZ, editors, *brandial'06 – Proceedings of the 10th Workshop on the Semantics and Pragmatics of Dialogue*, pp. 106–113, Universitätsverlag Potsdam, Potsdam.

Andy LÜCKING, Hannes RIESER, and Marc STAUDACHER (2006b), SDRT and Multi-modal Situated Communication, in David SCHLANGEN and Raquel FERNÁNDEZ, editors, *brandial'06 – Proceedings of the 10th Workshop on the Semantics and Pragmatics of Dialogue*, pp. 72–79, Universitätsverlag Potsdam, Potsdam.

David MCNEILL (1992), *Hand and Mind. What Gestures Reveal about Thought*, University of Chicago Press, Chicago.

David MCNEILL (2005), *Gesture and Thought*, University of Chicago Press, Chicago.

Richard MONTAGUE (1988), The Proper Treatment of Quantification in Ordinary English, in Jack KULAS, James H. FETZER, and Terry L. RANKIN, editors, *Philosophy, Language, and Artificial Intelligence*, volume 2 of *Studies in Cognitive Systems*, pp. 141–162, Springer Netherlands, ISBN 978-94-010-7726-2,

doi:10.1007/978-94-009-2727-8_7,
http://dx.doi.org/10.1007/978-94-009-2727-8_7.

Cornelia MÜLLER, Jana BRESSEM, and Silva H. LADEWIG (2013), Towards a grammar of gesture – a form based view, *Body–Language–Communication: An International Handbook on Multimodality in Human Interaction. (Handbooks of Linguistics and Communication Science 38.1)*, pp. 707–733.

Stephan OEPEN (2001), [incr tsdb()] — Competence and Performance Laboratory. User Manual, Technical report, Computational Linguistics, Saarland University, Saarbrücken, Germany.

Stephan OEPEN, Klaus NETTER, and Judith KLEIN (1997), TSNLP — Test Suites for Natural Language Processing, in John NERBONNE, editor, *Linguistic Databases*, pp. 13–36, CSLI Publications, Stanford, CA.

Patrizia PAGGIO and Costanza NAVARRETTA (2009), Integration and representation issues in the annotation of multimodal data, in Costanza NAVARRETTA, Patrizia PAGGIO, Jens ALLWOOD, Elisabeth ALSÉN, and Yasuhiro KATAGIRI, editors, *Proceedings of the NODALIDA 2009 workshop Multimodal Communication — from Human Behaviour to Computational Models*, volume 6, pp. 25–31, Northern European Association for Language Technology (NEALT).

Thies PFEIFFER, Florian HOFMANN, Florian HAHN, Hannes RIESER, and Insa RÖPKE (2013), Gesture Semantics Reconstruction Based on Motion Capturing and Complex Event Processing: a Circular Shape Example, in *Proceedings of the SIGDIAL 2013 Conference*, pp. 270–279, Association for Computational Linguistics, http://aclweb.org/anthology/W13-4041.

Livia POLANYI (1985), A Theory of Discourse Structure and Discourse Coherence, in *Proceedings of the 21st Meeting of the Chicago Linguistics Society*, Chicago, Illinois: Linguistics Department, University of Chicago.

Uwe REYLE (1993), Dealing with Ambiguities by Underspecification: Construction, Representation and Deduction, *Journal of Semantics*, 10:123–179.

I. A. SAG and T. A. WASOW (1999), *Syntactic Theory: A Formal Introduction*, Center for the Study of Language and Information, Stanford, California, ISBN 1575861615 (hard cover), 1575861607 (paper).

Mark STEEDMAN (2000), *The Syntactic Process*, The MIT Press.

Francis & Mark Turner STEEN (2013), Multimodal Construction Grammar, *Language and the Creative Mind*, pp. 255–274.

Inferring inflection classes with description length

Sacha Beniamine[1]*, Olivier Bonami*[1]*, and Benoît Sagot*[2]
[1] Université Paris Diderot, Laboratoire de linguistique formelle
[2] Inria

ABSTRACT

We discuss the notion of an *inflection class system*, a traditional ingredient of the description of inflection systems of nontrivial complexity. We distinguish systems of *microclasses*, which partition a set of lexemes in classes with identical behavior, and systems of *macroclasses*, which group lexemes that are similar enough in a few larger classes. On the basis of the intuition that macroclasses should contribute to a concise description of the system, we propose one algorithmic method for inferring macroclasses from raw inflectional paradigms, based on minimisation of the description length of the system under a given strategy of identifying morphological alternations in paradigms. We then exhibit classifications produced by our implementation on French and European Portuguese conjugation data and argue that they constitute an appropriate systematisation of traditional classifications. To arrive at such a convincing systematisation, it was crucial for us to use a *local* approach to inflection class similarity (based on pairwise comparisons of paradigm cells) rather than a *global* approach (based on the simultaneous comparison of all cells). We conclude that it is indeed possible to infer inflectional macroclasses objectively.[1]

Keywords:
morphology,
MDL,
inflection classes

[1] Work reported here has been presented at the First Quantitative Morphology Meeting (Belgrade, June 2015), at the 9th *Décembrettes* conference (Toulouse, December 2015), and at workshops organized by Université Paris Diderot and Labex EFL. We thank the audiences at these events and three anonymous reviewers for their comments. This work was partially supported by a public grant overseen by the French National Research Agency (ANR) as part of the "Investissements d'Avenir" program (reference: ANR-10-LABX-0083).

INTRODUCTION

The concept of INFLECTION CLASS is central to many analyses of inflection systems, both in theoretical linguistics (see among many others Matthews 1972; Carstairs 1987; Wurzel 1989; Aronoff 1994; Dressler and Thornton 1996; Corbett 2009) and in psycholinguistic studies (see among others Milin *et al.* 2009; Veríssimo and Clahsen 2014). Inflection class systems are commonly taken to be a classification of lexemes according to their INFLECTIONAL REALISATIONS. While such a broad characterization is largely agreed upon, there are alternative ways of applying it. Some authors (e.g. Stump and Finkel 2013) insist on strict identity of inflectional realisations, leading to systems with a large number of small classes. Many others follow traditional descriptions in defining a small number of classes based on broad similarity, and allowing some amount of variability within each class. Despite this uncertainty as to the characterization of classes a partition of the lexicon into classes is often taken for granted as a starting point for analysis, rather than explicitly argued for.

In this paper, we show that inflection classes can be deduced in a systematic and motivated way from raw paradigms, without introducing any preconception about organizing principles other than similarity. Our approach is abstractive in nature (in the sense of Blevins 2006), and is intended to systematize the strategies of descriptive morphologists in finding inflection classes.

The strategy is systematic enough to allow for full computational implementation.[2] We use the minimum description length principle (Rissanen 1984) to balance similarity within classes and dissimilarity between classes. This presupposes that we have a way of assessing similarity between overall inflection patterns. In this paper, we will consider two different but closely related ways of assessing that similarity. Under a GLOBAL approach, inflection patterns are determined by comparing all of the inflected forms of a lexeme simultaneously; under a LOCAL approach, the overall characterization is deduced from pairwise comparisons of paradigm cells. We propose a simple procedure for identifying patterns that can be applied either locally or globally,

[2] The full code to replicate the classifications discussed in this papier is available at `http://drehu.linguist.univ-paris-diderot.fr/qumin/`.

and show that a local approach captures the kinds of generalizations that descriptive morphologists rely on for classification.

The paper is organized as follows. First, we explore alternative definitions of inflectional classes and inflectional realisations. Then, we present a strategy for inferring inflection classes from raw paradigms in two steps: deducing inflectional realisations from the forms, and classes from realisations. In the third section, we present the detailed algorithms devised to perform each of these two steps, and describe the description length measure we use. The final section discusses results on both French and European Portuguese.

1 WHAT ARE INFLECTION CLASSES?

In this section, we argue that the apparent consensus on inflectional classification masks important differences between accounts that often rest on unstated theoretical assumptions, especially the role given to morpho-phonology, the basic units posited by the model, segmentation strategies, and the definition of similarity. For this reason, there is no agreed upon method to rigorously infer the classes from raw paradigms.

1.1 *Two definitions of inflection classes*

Following Aronoff (1994, p. 64) and Carstairs-McCarthy (1994, p. 639), we could define an inflection class as "a set of lexemes whose members each select the same set of inflectional realisations". We illustrate the definition with the twelve classes of Latin nouns in Table 1, as presented by Stump and Finkel (2013).

According to this definition, an inflection class system is an exhaustive partition of the set of lexemes in several non-overlapping classes. All members of a class have the exact same inflectional behavior. For example, classes (2a) and (2b), although they share all their other realisations, are distinct in that (2b) shows no affixal realisation for nominative and vocative singular.

While they match both Aronoff and Carstairs-McCarthy's definition of inflection classes, the 12 inflection patterns identified as rows in Table 1 do not correspond to the traditional characterisation of the Latin system. The tradition distinguishes only five classes, which group together some of the rows. Within those classes, as Dressler *et al.*

Table 1: Latin noun endings organized by declensions

		Singular						Plural					
Declension		NOM	VOC	ACC	GEN	DAT	ABL	NOM	VOC	ACC	GEN	DAT	ABL
First	(1)	a	a	am	ae	ae	ā	ae	ae	ās	ārum	īs	īs
Second	(2a)	us	e	um	ī	ō	ō	ī	ī	ōs	ōrum	īs	īs
	(2b)	–	–	um	ī	ō	ō	ī	ī	ōs	ōrum	īs	īs
	(2c)	um	um	um	i	ō	ō	a	a	a	ōrum	īs	īs
Third	(3a)	s	s	em	is	ī	e	ēs	ēs	ēs	um	ibus	ibus
	(3b)	–	–	–	is	ī	e	a	a	a	um	ibus	ibus
(i-stems)	(3c)	is	is	em	is	ī	e	ēs	ēs	ēs	ium	ibus	ibus
	(3d)	s	s	em	is	ī	e	ēs	ēs	ēs	ium	ibus	ibus
	(3e)	e	e	e	is	ī	ī	ia	ia	ia	ium	ibus	ibus
Fourth	(4a)	us	us	um	ūs	uī	ū	ūs	ūs	ūs	uum	ibus	ibus
	(4b)	ū	ū	ū	ūs	ū	ū	ua	ua	ua	uum	ibus	ibus
Fifth	(5)	ēs	ēs	em	ēī	ēī	ē	ēs	ēs	ēs	ērum	ēbus	ēbus

This table follows Stump and Finkel (2013, p. 183, Table 7.1). We omit the locative, reorder the paradigm cells and add numbering to facilitate reference to specific declensions.

(2008, p. 52) remind us, "not all nouns of one class inflect in exactly the same way". For example, while some lexemes of the third declension end in *-ium* in the genitive plural (3c, 3d, 3e), others end in *-um* (3a and 3b). Rather than identity, then, members of the traditional classes display a strong degree of similarity.

This example is representative of a general observation that traditional descriptions of inflection systems distinguish a small number of broad classes, comprising both common patterns seen as regular and less common patterns seen as deviating from the regular situation. This leads some authors, such as Brown and Hippisley (2012, p. 4), to adopt a less strict criterion in the definition of inflection classes, which are then seen as "classes of lexemes that share similar morphological contrasts".

The existence of two alternative definitions of inflection classes is sometimes the source of confusion. For instance, it is notable that, after proposing a definition of classes based on identity of realisation,

Carstairs-McCarthy's (1994)'s account of Latin nouns relies on mere similarity: starting from six classes (out of 8), he proposes to merge some of them so as to have a system of three classes. Whatever one may think of the motivation for such merges, the absence of a clear distinction between the two notions of inflectional classification makes such proposals hard to evaluate.

In a series of influential publications, Dressler and colleagues (Dressler *et al.* 1987; Dressler and Thornton 1996; Kilani-Schoch and Dressler 2005) propose not to choose between the two strategies and provide separate names for the two types of classes: MICROCLASSES are small uniform classes whose members have identical realisations, while MACROCLASSES are large classes exhibiting some amount of internal variation.[3]

Dressler and Thornton (1996) define microclasses and macroclasses as the two extremes in an inflection class hierarchy which may also contain classes of intermediate grain. This is very similar in spirit, if not in the details of execution, to inflection class hierarchies customarily used in Network Morphology (Corbett and Fraser 1993; Brown and Hippisley 2012).

Under this view, microclasses correspond to Aronoff and Carstairs-McCarthy's definitions. There is little doubt that, given a set of paradigms and some way of abstracting inflectional realisations from the paradigms, one can deduce a unique system of microclasses appropriately describing the system. The situation of macroclasses is more uncertain. Given that macroclasses are defined in terms of similarity, and that similarity is a gradual and multidimensional notion, there are various ways to partition a system into macroclasses, among which it is not obvious which should be chosen, short of a quantitative evaluation of the complexity of the resulting grammar (Walther 2013). For instance, there is no obvious way of deciding whether the Latin first and second declensions should be considered to form one class

[3] Our use of the term 'microclass' differs from that of Dressler and coauthors in one minor way. For Dressler and Thornton (1996), "An isolated paradigm is a paradigm which differs morphologically or morphophonologically from all other paradigms; it does not form a microclass of its own but is considered a satellite to the most similar microclass." In our usage, isolated paradigms are just microclasses of cardinality 1. This is of little theoretical consequence, but dramatically increases the number of microclasses for some systems.

(as in Carstairs-McCarthy 1994), because they inflect similarly in the dative, ablative, and locative plural, or two, because the realisations are distinct everywhere else in the paradigm.

Since the validity of microclasses is not under question, we will focus our attention on the status of macroclasses. We will take a system of inflection classes to be a partition of lexemes into classes, and attempt to infer a system of macroclasses from observed paradigms. Notice that we focus on the inference of macroclasses rather than a full hierarchy of classes of variable granularity. While this is definitely an interesting endeavour (see for instance Brown and Evans 2012; Lee and Goldsmith 2013; Bonami 2014 for some proposals), it calls for a different methodology, and does not directly help us evaluate which partition in the hierarchy should correspond to the level of macroclasses.

Defining macroclasses requires a definition of inflectional realisations from which the similarities follow, and a criterion to decide the appropriate level of generality. In the following two sections, we first describe some possible ways of defining inflectional realisation before investigating the possible criteria with which to define macroclasses.

1.2 *Macroclasses follow from inflectional realisations*

1.2.1 Circularity of inflectional realisation definitions

Any enterprise in inflectional classification starts with the identification of inflectional realisations. The heuristics used for that purpose are seldom made explicit, although they are rarely obvious. For instance, it is customary to assume that inflectional variability combines the use of different patterns of stem allomorphy and different affixal exponents, although deciding on the exact boundary between stem and exponent is far from being a trivial matter. Carstairs-McCarthy's (1994)'s work on inflection classes is commendable for its explicitness in such matters. We thus propose to explore it in some detail.

Carstairs-McCarthy's strategy relies on two central decisions. First, inflection classes are defined purely in terms of affixal exponence, and abstract away from stem allomorphy. Thus inflectional realisations are considered affixes, and any alternation that is not affixal is ignored. Second, segmentation choices are justified by the desirability of the inflection class system they yield. This is motivated by the goal of testing whether inflection class systems satisfy the 'No Blur

Table 2: Latin masculine declensional endings without and with thematic vowels

	Singular						Plural			
Declension	NOM	VOC	ACC	GEN	DAT	ABL	N/V	ACC	GEN	D/A
First	a	a	am	ae	ae	ā	ae	ās	ārum	īs
Second	us / –	e	um	ī	ō	ō	ī	ōs	ōrum	īs
Third (cstem)	s / – / ēs	s / – / ēs	em	is	ī	e	ēs	ēs	um	ibus
(mixed)	s	s	em	is	ī	e	ēs	ēs	ium	ibus
(istems)	is	is	im > em	is	ī	ī > e	ēs	īs > ēs	ium	ibus
Fourth	us	us	um	ūs	uī	ū	ūs	ūs	uum	ibus

	Singular						Plural			
Declension	NOM	VOC	ACC	GEN	DAT	ABL	N/V	ACC	GEN	D/A
First	–	–	m	ī	ī	V:	ī	:s	:rum	īs
Second	s / –	e	m	ī	V:	V:	ī	:s	:rum	īs
Third (cstem)	s / –	s / –	m	s	ī	e	ēs	ēs	um	bus
(mixed)	s	s	m	s	ī	e	ēs	ēs	um	bus
(istems)	s	s	m	s	ī	V: > e	ēs	:s > ēs	um	bus
Fourth	s	s	m	s	ī	V:	:s	:s	um	bus

This table is adapted from Carstairs-McCarthy (1994, pp. 749-750). The symbol > means 'tends to be replaced by'. Slashes separate affixes which are distributed on a partly phonological, partly arbitrary basis. In each column, shades of gray highlight repeated affixes when they violate the No Blur Principle.

Principle', according to which any affix realising some paradigm cell must be either a class identifier (i.e. specific to that class) or a class default (i.e. common to all those classes that do not possess a class identifier). By Carstairs-McCarthy's reasoning, if a system can be described with classes that satisfy the No Blur Principle, then that classification should be used, even if there are alternative classifications that do not satisfy the principle.

Carstairs-McCarthy explores two alternative segmentations of the affixes of masculine latin nouns, reproduced in Table 2. We show blur in columns using grayed cells. The traditional analysis, presented at the top of the table, presents some blur: for instance, in the dative plural, *-īs* is neither a class identifier (it is common to two classes) nor a default (since the other possible affix, *-ibus*, is not an identifier either). In an alternative analysis, presented at the bottom of the table, theme vowels are taken to be part of stems rather than affixes. This analysis

shows twice as many blurred columns. Therefore, Carstairs-McCarthy prefers the first analysis. Whatever one may think of the relative merits of the two analyses and the relevance of the No Blur Principle, it is worth noting that Cartairs-McCarthy's heuristic for choosing a segmentation leads to circularity: inflection classes are taken to be sets of words displaying the same set of inflectional realisations, but what counts as an inflectional realisation is decided on the basis of the desirability of the resulting inflection class system. Such circularity is particularly vivid in Carstairs-McCarthy's paper, but we suspect that it is present in many descriptions that do not make their segmentation heuristics explicit. This dependency of the realisations on the classes is problematic in the context of an abstractive approach to inflectional classification, where the realisations are the starting point for the inference of classes.

More generally, despite relevant attempts (e.g. Montermini and Boyé 2012; Spencer 2012), there is no agreed upon systematic strategy to decide where to place the boundary between stem and affix; and as Blevins (2005) and Blevins (2006) argues, in some systems, there is just no coherent way of making such a decision. From this observation we conclude that a systematic method for inferring inflectional realisations should not rely on a preexisting segmentation into stems and affixes. Given this, one possible way forward is to explore different segmentation strategies and rely on Occam's razor to decide which is optimal (Sagot and Walther 2011; Walther and Sagot 2011). Another, which we pursue here, is to take whatever alternation is seen in the data at face value, irrespective of how (un)systematic it is or whether it affects peripheral rather than central segments of the alternating forms.

1.2.2 Global and local alternation patterns

To avoid making any undermotivated decision as to the boundary between affixal exponence and stem allomophy, we define inflectional realisation in terms of the alternation patterns relating the different forms in the paradigm of a lexeme to each other. Interestingly, wherever paradigms have a more than two cells, there are at least two strategies for identifying such patterns. We illustrate this with the small sample of the French adjectival lexicon in section A of Table 3.

Table 3: Alternative segmentation choices for a subset of French adjectives

	A. Paradigms			B. Stem and exponents, global		
lexemes	M.SG	F.SG/PL	M.PL	M.SG	F.SG/PL	M.PL
NORMAL	nɔʁmal	nɔʁmal	nɔʁmo	Xal	Xal	Xo
VERT	vɛʁ	vɛʁt	vɛʁ	X	Xt	X
BLEU	blø	blø	blø	X	X	X
	C. Patterns, local			D. Patterns, global		
	M.SG ∼ F.SG/PL	M.SG ∼ M.PL	F.SG/PL ∼ M.PL	M.SG ∼ F.SG/PL	M.SG ∼ M.PL	F.SG/PL ∼ M.PL
NORMAL	X ∼ X	Xal ∼ Xo	Xal ∼ Xo	Xal ∼ Xal	Xal ∼ Xo	Xal ∼ Xo
VERT	X ∼ Xt	X ∼ X	Xt ∼ X	X ∼ Xt	X ∼ X	Xt ∼ X
BLEU	X ∼ X	X ∼ X	X ∼ X	X ∼ X	X ∼ X	X ∼ X

In this table, gray cells highlight patterns that are common to two lexemes, showing that only local patterns capture the simiarity between NORMAL and BLEU.

The first and most familiar strategy consists of identifying the similarities and differences between forms GLOBALLY. This is indicated for our toy example in section B of Table 3: in each row, the substring common to all paradigm cells has been replaced by a variable. An alternative strategy, often invoked by proponents of implicative approaches to morphology (e.g. Blevins 2005, 2006; Ackerman *et al.* 2009; Bonami and Beniamine 2015), consists of identifying LOCAL similarities between pairs of paradigm cells. This is indicated in section C of Table 3, where each column now corresponds to a different pair of cells. Note that since we are dealing with a mostly concatenative system, both strategies can be seen as amounting to proposing a segmentation of words into constant ('stems') and variable ('affixes') subparts. However, in the global approach the constant part is common to the whole paradigm, whereas in the local approach it is particular to one pair of cells: for instance, M.SG *normal* is segmented into *norm*+*al* for purposes of comparison with the M.PL, but not for purposes of comparison with the feminine.

One way of highlighting the difference between the two strategies is to tabulate the consequences of a global strategy for the description of alternations between pairs of cells. This is done in section D of Table 3, which just sums up the information in section B of Table 3 in the forms of relations between pairs of cells. One may note that, according to the local strategy, section C of Table 3, the adjective BLEU shares inflectional characteristics with both NORMAL and VERT: like NORMAL, it does not alternate between M.SG and F; like VERT, it does not alternate between M.SG and M.PL. By contrast, according to the global strategy, VERT and BLEU share the same characteristic of not alternating between the M.SG and M.PL, but NORMAL and BLEU do not have anything in common.

If binary alternation patterns are the inflectional realisation, then microclasses are defined by vectors of patterns, where each coordinate of the vector indicates the pattern instantiated in that microclass for a different pair of paradigm cells. These vectors are represented by rows in sections C and D of Table 3. We thus conclude that in our toy example, the global and local strategies give rise to the same microclasses. However, relations of similarity among these microclasses are different. Hence the use of local or global inflection patterns to char-

acterise inflectional realisation may influence what macroclasses will be inferred.

One of the goals of this paper is to evaluate the relative perspicuity of inflectional classifications based on local and global alternation patterns. For the time being, let us comment briefly on the relationship between alternation patterns, whether global or local, and segmentation of words into stems and affixes. There is a natural relation between global patterns and stem-based segmentation. Since global patterns identify a constant subpart common to the whole paradigm, in the context of concatenative morphology, a global pattern corresponds to an analysis where each lexeme is constrained to using a single stem, and any variable element is taken to be affixal material. Interestingly, there is no such clear relation between local patterns and the classical notion of a stem. As we highlighted above, one and the same word filling one paradigm cell may be segmented differently for the purposes of comparison with two other cells. Hence, under a local pattern view, even individual paradigm cells are not associated with a unique constant substring which could be identified as a stem.

1.3 *Criteria for macroclasses*

In the preceding section we showed how different strategies for describing inflection systems, be they based on segmentation between stems and affixes or on alternation patterns, lead to different classifications. We now turn to the problem of deciding which groupings of microclasses should be considered as forming a single macroclass. We explore five strategies found in the literature: using an ad-hoc combination of criteria, the regular/irregular distinction, maximisation of inflection class heterogeneity, maximisation of internal predictability, and maximisation of descriptive economy.

1.3.1 Ad-hoc criteria

Descriptive morphologists usually motivate their classification highlighting some property or set of properties which the classes happen to differ in. For instance, Latin verb classes are characterised by the quality and length of the theme vowel in the present active infinitive: *-ā-* in the first conjugation, *-ē-* in the second, *-e-* or *-i-* in the third, and *-ī-* in the fourth. Of course, this is not the only way in which Latin conjugations contrast, and not all forms exhibit such a contrast. As any

description of Latin conjugation will note when commenting on the third conjugation, some verbs in that class have an indicative present active 1SG form similar to that of a first conjugation verb, cf. SECŌ 'cut' (INF *secāre*) vs. SERŌ 'sow' (INF *serere*); others do contrast with the first conjugation in that paradigm cell, but fail to contrast with the third conjugation, cf. CAPIŌ 'take' (INF *capere*) vs. SAEPIŌ 'surround' (INF *saepīre*). Full classification relies on an ordering of highlighted *ad-hoc* properties: in the case of Latin, tradition holds that contrasts in the infinitive are more important than contrasts in the indicative present first person singular.

There are two concerns with such a strategy for motivating a classification. First, it is unclear whether the highlighted properties are selected *post-hoc* to contrast pre-established classes, perhaps for pedagogical purposes, or whether they really play a distinguishing role. In the case at hand, it seems arbitrary that the infinitive is used to motivate the distinction between the four classes when the relevant contrast is also apparent e.g. in the present 1PL. Second, it is unclear that there is any strong motivation for the way the contrasts are prioritised.

The situation just discussed in the case of the traditional classification of Latin verbs also holds for more elaborate, thoughtful, and theoretically-informed classification attempts. We exemplify this situation by discussing , in some detail, the proposed classification of French verbs by Kilani-Schoch and Dressler (2005).

As we saw before, in Natural Morphology, macroclasses are viewed as the top-level partition in an inflection class tree (Dressler *et al.* 2008; Kilani-Schoch and Dressler 2005; Dressler and Thornton 1996). In these accounts, Macroclasses, just as classes of all granularities, are defined by implicational PARADIGM STRUCTURE CONDITION (PSCs). To study the nature of PSCs, we reproduce below those presented in Kilani-Schoch and Dressler (2005) for some classes of French verbal inflection.

(1) Macroclass I:

$$\text{Infinitive } /X+e/ \Rightarrow \begin{cases} \text{Past Participle} & = /X+e/ \\ \text{Simple Past first person} & = /X+e/ \\ \text{Singular present} & = /X/ \\ \text{Indicative present 3rd plural} & = /X/ \\ \text{Subjunctive present} & = /X/ \end{cases}$$

(2) Class I.1:

Imperfect [parl + ɛ], future [parl + ər + e].

(3) Class II.2:

$$\text{Infinitive /Xwar/} \Rightarrow \begin{cases} \text{Past Participle in /y/} \\ \text{Simple Past in /y/} \\ \text{by default, /wa/ is part of the infinitive} \end{cases}$$

We first remark that PSCs are of variable nature. They are sometimes formulated as implicative relations (Wurzel 1984; Ackerman *et al.* 2009; Stump and Finkel 2013), as is the case for macroclass I of French verbs reformulated in (1) or in class II.2 as shown in (3). These implications are sometimes relationships between two cells (if some cell is X, then some cell is Y), as in (1), and sometimes between a cell and an abstract segmented unit as in (3). Some subclasses, on the contrary, are defined by the exponence strategies they implement, as in (2) for microclass I.1.

In Kilani-Schoch and Dressler (2005)'s analysis of French verbs, the implications are frequently true for all the other classes. For example, the antecedent of the implication in (1), having an infinitive in /Xe/, is only true of the verbs in macroclass I. As a consequence, all the implications based on this premise are true of the whole system. What is implicitly defining that macroclass, then, is not the PSC but the exponent: macroclass I is the class of all verbs with an infinitive ending in /e/. The same could be said of the PSC from (3) which is true of the whole system because only verbs of the class II.2 share an infinitive ending in /-war/, revealing that it is in fact defined not by the implication but by the ending. We conclude then that, while PSCs are formulated as implications, classes are really defined by exponence strategies, mostly with a focus on the infinitive.

In light of these observations, it appears that a class is sometimes a set of lexemes having one or more common exponents (as we showed for I and II.2), sometimes a set of lexemes for which some implicative relationship between cells hold. Since macroclasses are motivated by different types of criteria, we cannot assume that they are consistently the same kind of object. If one chooses to keep both types of criteria, it is not clear how one should decide which to apply when. It seems preferable to build a class system relying only on one criterion.

1.3.2 External motivation: regularity

Another organizing principle is at work in Kilani-Schoch and Dressler's (2005)'s classification of French verbs. A core assumption of that work is a dual mechanism approach to inflection processing (see Clahsen 2006 and references therein), according to which (i) there is a categorical distinction between regular and irregular lexemes, and (ii) regular and irregular lexemes are processed differently by speakers. Whether a lexeme is regular or irregular cannot be established by examination of the synchronic inflection system, but only through assessments of productivity (only regular patterns are deemed productive) or psycholinguistic experimentation (regular and irregular lexemes should lead to measurably different learning, processing, and production). Kilani-Schoch and Dressler hold that the contrast between regulars and irregulars should be the principal criterion to distinguish macroclasses. Hence their classification makes a main distinction between two macroclasses, corresponding to the traditional first conjugation (infinitives in *-er*) vs. all other verbs.

Whatever one may think of the merits of the dual mechanism hypothesis or of the assumption that regularity in French holds only of the traditional first conjugation (see Bonami *et al.* 2008), the important point for present purposes is that Kilani-Schoch and Dressler's criterion for macroclasses is fundamentally different from the criterion used to group lexemes into microclasses. Macroclasses are No longer a generalisation over microclasses, but rather a completely different classification of lexemes, whose empirical validity cannot be established by examination of the internal structure of the synchronic system. Again, while this is a defendable position, it is unclear why one type of criterion should be privileged over another. Evidently there are multiple ways of classifying lexemes that may be relevant for different purposes, and it is not clear that there is merit in attempting to combine all such classifications in a single tree. In particular, it is an open question how exactly a broad classification based on structural similarity and contrast between inflection patterns correlates with contrasts in productivity and/or ease of processing. Presupposing a strong association between the two does not help explore the issue.

In the remainder of this paper, we will focus on approaches to inflectional classification that rely solely on examination of similarity and differences between paradigms.

1.3.3 Heterogeneity among classes

In the context of defining a canonical typology of inflection class systems, Corbett (2009, p. 4) formulates two important criteria for canonical inflection classes, respectively on distinctiveness and cohesion of classes:[4]

(4) a. "Criterion 1: In the canonical situation, forms differ as consistently as possible across inflectional classes, cell by cell."

b. "Criterion 3: Within a canonical inflectional class each member behaves identically."

According to Corbett, a canonical inflection class system is a single partition of the set of lexemes where each class is maximally cohesive internally and maximally distinct from other classes. Interestingly, Criterion 3 is reminiscent of the definition of micro-classes. It is tempting then to assume that macro-classes are defined by Criterion 1: macro-classes should be strikingly different from one another. This seems to match traditional practice, and leads to the satisfactory conclusion that a canonical system is a system where micro-classes and macro-classes coincide.

While Criterion 1 definitely captures part of the intuition behind macro-classes, we should be wary of not applying it too strictly. In any system where one paradigm cell inflects uniformly, all lexemes share at least one inflectional realisation, and this common inflectional realisation forbids perfect heterogeneity between classes. As a consequence, there is no partition that maximises distinctiveness, and hence no macroclass other than the system as a whole. Such a definition of macro-classes would then be too dependent on a rather unilluminating property of the system. Moreover, maximisation of distinctiveness does not strictly match traditional practice either. For instance, in the case of Latin nouns (1), it is not usual to suggest fusing the third and fifth declensions, despite the fact that they share the exponent *-e* in the singular ablative.

[4] Corbett's Criterion 2 refers to the shape of paradigms, and does not directly concern us here.

We thus conclude that while distinctiveness is an important property of macro-classes, it cannot be used as the sole criterion for choosing which partition should count as a partition into macro-classes.

1.3.4 Predictability within classes

Going back to Carstairs-McCarthy (1994), we find that he justifies the merging of classes into what he calls macroclasses when different affixes can be seen as suppletive allomorphs predictable from some other phonological or morphological factor (they are not competing for the speakers) (see Table 4).

This leads him to merge the first and second Latin declension (see Table 2), despite their strong dissimilarity. Indeed, the first two declensions are mostly predictable on the basis of gender. In the same way, some variations of the 3rd declension are predictable on the basis of phonological properties of the stem. These are indicated by a swung dash in Table 4. This is contrary to the intuition that macroclasses are classes of lexemes that inflect alike.

In addition, some alternations indicated by a slash in Table 4 do not correspond to systematic alternations. In this case, the classes are merged together because of the similarity of their paradigms, not because of their predictability.

Table 4: Table from Carstairs-McCarthy (1994, p. 751)

			Singular			
Declension	NOM	VOC	ACC	GEN	DAT	ABL
First/Second	a ∼ us / –	a ∼ e	am ∼ um	ae ∼ ī	ae∼ ō	ā ∼ ō
Third	s ∼ – / ēs ∼ is	s ∼ – / ēs ∼ is	em ∼ im	is	ī	e
Fourth	us	us	um	ūs	uī	ū
			Plural			
	NOM/VOC	ACC	GEN	DAT/ABL		
First/Second	ae ∼ ī	ās ∼ ōs	ārum ∼ ōrum	īs		
Third	ēs	ēs	um ∼ ium	ibus		
Fourth	ūs	ūs	uum	ibus		

Original caption: “Latin masculine nouns: third analysis, designed to remove blur. Forms separated by a swung dash are to be understood as distributed on the basis of gender (in the 1st/2nd declension) or of phonological characteristics of the stem. The distribution of forms separated by a slash is not governed in this way.”

Beyond the specific predictors used by Carstairs-McCarthy, we can see that merging paradigms according to predictability or similarity of the inflectional realisations leads to different results. Moreover, it is expected that merging together very similar paradigms is not favorable to prediction. Let us take paradigm entropy (Ackerman and Malouf 2013), the average conditional entropy of one paradigm cell given another paradigm cell, as a measure of internal predictability in a class. Unisng paradigm entropy, it becomes apparent that in fact, merging similar classes hinders predictability rather than helping it. In the case at hand, merging (2a) and (2b) in Table 1, which only differ by nominative and vocative singular, raises the difficulty of predicting these cells from any of the others, as having an accusative in *-um* and knowing that the noun is of the second declension will not guarantee that one can guess the correct nominative form. A macroclass comprising (2a) and (2b) would be justified if macroclasses are taken as similarity-based classes, but not if they are taken as classes with low paradigm entropy. On the other hand, one would not want to merge (1a) and (2a) on the basis of similarity. However, since they share few realisations, merging them would not raise the class paradigm entropy much. For example, from the accusative form, two patterns would be available to form the nominative, either *-am* → *-a* or *-um* → *-us*. This, however, does not make prediction more difficult, as only accusative forms ending in *-am* are candidates for the first pattern, and those ending in *-um* for the second one.

Devising an entire classification of macroclasses in a way that minimises the paradigm entropy in each class would lead to classifications that differ very strongly from what descriptive linguists produce. In this paper, we will rather try to find macroclasses on the basis of similarity. However, we should remember that those classes are not expected to have a lower paradigm entropy than the whole system.[5]

[5] Given several competing analyses of a system into classes on the basis of their realisations, one could prefer that which conveniently predicts other grammatical features. Corbett (1982) has argued that it is preferable to define four macroclasses of Russian nouns, rather than the three traditionally recognized, as it offers a better predictibility of gender. As a first step towards automatic inference of inflectional classification, the current study bases the inference of macroclasses strictly on wordforms. However, the model could be extended in a straightforward manner to cluster classes on the basis of other features in addi-

1.3.5 Maximisation of descriptive economy

Another approach to the problem of choosing how to define macroclasses relies on the idea that, in theory, the optimal set of macroclasses should result in the most economical description of the morphological system as a whole. This idea has been explored in particular by Sagot and Walther (Sagot and Walther 2011; Walther and Sagot 2011; Walther 2013; Sagot and Walther 2013; Walther 2016), who compare manually crafted descriptions, comprising a morphological grammar and a morphological lexicon, using a quantitative measure of their descriptive economy based on the information-theoretic notion of DESCRIPTION LENGTH (Rissanen 1978). Such an approach allowed them to compare competing accounts of a number of morphological (sub)systems in a variety of languages (French, Maltese, Khaling, and Latin), based on grammars implemented in the Alexina framework, an implementation of Walther's morphological formalism, for which see now (Walther 2016). These competing accounts can vary in different ways, one of which being the inventory of macroclasses, which roughly correspond to what they refer to as inflection patterns. For instance, Sagot and Walther (2011) compares the description lengths of four descriptions of French verbal inflection that contrast in the number of macroclasses they distinguish (from 1 to 139), in relation with different ways to dispatch morphological information between the grammar and the lexicon (e.g. lexically specified stem suppletion vs. stem alternation patterns encoded in the grammar).

While Sagot and Walther's work is an important inspiration for the strategy we will develop later in this paper, there are two fundamental limitations of their work. First, the fact that they rely on a specific description formalism to encode all competing accounts inevitably biases and reduces the set of possible accounts that can be compared. Second, and more importantly, they only compare a handful of manually crafted grammars. Without a way to systematically explore the space of possible descriptions, they can only draw conclusions from the relative compactness of the competing descriptions they compare.

tion to alternation patterns. We leave the exploration of such a possibility to a future study.

To conclude this section, we have argued that a coherent definition of macroclasses should rely on a single, well-conceived criterion to assess the level of accepted similarity. Several competing criteria are sometimes used to define macroclass membership, and most criteria used in the literature rely on more than the forms and inflectional realisations themselves. In this work, we ask whether macroclasses can be inferred from the sole examination of paradigms. This has the advantage that any preconceived idea about other properties that macroclasses have can be tested empirically. For example, we will be able to observe if we find only two macroclasses that conform to the categorical regular/irregular contrast presupposed by a dual mechanism approach to morphological processing.

2 INFERRING INFLECTION CLASSES

To automatically infer macroclasses from paradigms of raw forms, we take on two tasks, treated sequentially. First, given paradigms of forms, we want to infer all relevant alternation patterns following either a local or a global segmentation. The two segmentation strategies need to be strictly comparable. Second, given a table of alternation patterns, we attempt to infer micro- and macroclasses in a principled way.

2.1 *From forms to patterns*

The first task at hand is to infer alternation patterns from surface forms. We first describe previous work on the subject, then describe our algorithm.

2.1.1 Previous work on inflectional rule inference

A substantial amount of work has already been done on automatic inference of inflection rules from inflected forms, either in the context of modeling a speaker's knowledge of inflection (Albright and Hayes 2003, 2006) or in a Natural Language Processing context, with the goal of expanding sparse lexica (Durrett and DeNero 2013; Ahlberg *et al.* 2014; Nicolai *et al.* 2015). In this section, we review relevant aspects of these attempts.

Given a set of forms, one can formulate a large number of alternation patterns relating them. Choosing an appropriate function is an optimisation problem, seeking to minimise both the total number of

Table 5: Illustration of the alignment problem for two imaginary languages

(a) Infix language

SG	PL
to	bato
ri	bari
su	basu
ne	bane
ba	baba

(b) Prefix language

SG	PL
to	tabo
ri	rabi
su	sabu
ne	nabe
ba	baba

(c) Alignments of *baba* ∼ *ba*

Alignment						Pattern SG ∼ PL
PL	b	a	b	a		
(i) SG	_	_	b	a		_ ∼ ba_
(ii) SG	b	a	_	_		_ ∼ _ba
(iii) SG	b	_	_	a		_∅_ ∼ _ab_

patterns postulated to describe a system and to maximise the morphophonological naturality of the function. To explore the problem, let us consider two imaginary languages marking the opposition between singular and plural nouns as indicated in Table 5.

The two languages share exactly one lexeme, whose singular form is *ba* and whose plural is *baba*. There are a number of alternative ways of conceiving of the exponent of plural for that morpheme. Three prominent possibilities are (i) a *ba-* prefix, (ii) a *-ba* suffix, or (iii) an *-ab-* infix.[6] To these three possibilities correspond the three patterns listed in section (c) of Table 5, which in turn correspond to three ways of aligning the two forms. These toy languages are designed to highlight the fact that the choice of a pattern for a given lexeme is dependent on what happens in the rest of the language. In the context of language (a), where all other nouns mark the plural by prefixing *ba-*, it is clearly preferable to adopt a prefixation analysis (i); on the other hand, in the context of language (b), where all other nouns mark the plural by infixation, no descriptive linguist would doubt that the appropriate analysis for *ba*∼*baba* is an infixation analysis.

The task of deciding which alternation pattern is most relevant to relate two forms usually requires at least two steps: choosing an alignment, and abstracting a pattern from that alignment. The ambiguity can be resolved at the alignment stage by finding only one alignment or once all possible patterns are known. Note that the local and global strategies described above differ in how they perform the alignment step.

Extant approaches contrast in the way they deal with these issues. First, Durrett and DeNero (2013) infer global segmentations via

[6] Further possibilities include reduplication of the initial or final syllable.

the alignment of all forms to a base form. Ahlberg *et al.* (2014) directly align all forms of a paradigm together, also performing a global segmentation. On the other hand, Albright and Hayes (2003) and Albright and Hayes (2006) explicitly model local alternation patterns. Nicolai *et al.* (2015) compare forms locally, but only include pairs containing a designated base form, and thus do not take into account the whole array of possible alternations. Second, Durrett and DeNero (2013) and Albright and Hayes (2006) both use string alignment algorithms based on edit distance. The former perform iterated alignments to make their algorithm paradigm aware (which is why their alignment is global) while the latter optimise the similarity of aligned segments in terms of phonological features. Ahlberg *et al.* (2014) rely on transducer intersection to find the optimal alignment, and Nicolai *et al.* (2015) use the Expectation-Maximisation algorithm to learn atomic operations rather than entire alignments.

Although these studies are important sources of inspiration for the algorithm presented below, the strategies they implement are not quite appropriate for our current goals. The use of a privileged base form makes sense when trying to fill sparse paradigms as did both Durrett and DeNero (2013) and Nicolai *et al.* (2015): picking a frequent base form then allows one to reliably make inferences even for infrequent lexemes. However, while some forms can be prominent on the basis of informativeness, markedness, or other factors,here is no *a priori*otivation for favouring a base form in the identification of inflection classes. Speakers may be initially exposed to any form of a lexeme, and are able to draw inferences about the rest of that lexeme's paradigm on that basis, exhibiting no dependency on a designated base (Ackerman *et al.* 2009; Bonami and Beniamine 2016).

Likewise, Albright and Hayes's Minimum Generalisation Learner has a crucial property: the patterns it finds are gradually generalised, and generalisations at all levels are remembered. This is crucial to modeling the phenomenon of *Islands of reliability*, whereas lexemes that are phonotactically more typical of an inflection pattern are more strongly associated by speakers with that pattern. For our purposes though, it is crucial that each pair of form be associated with a single pattern, so that the lexicon is partitioned according to which pattern each lexeme instantiates. In addition, not having to keep track of all intermediate generalisations considerably reduces the algorithmic

complexity of the task, an important practical consideration when our experiments will rely on comparisons of thousands of pairs of cells for thousands of lexemes.

Finally, none of the studies we review here provide an algorithm allowing for the comparison between global and local strategies. We thus devise one that allows for strict comparison of both strategies.

2.1.2 Our pattern algorithm

To compare local and global segmentation strategies, we devise a segmentation process with two minimally different variants, which both output exactly one pattern per pair of cells. We use the same algorithm in both cases, changing only the number of forms we input.

We exemplify the algorithm on a sub-paradigm of the French verb AMENER 'bring', consisting of the three indicative present plural forms, and start with the global strategy. In that context, all forms of a paradigm are input at once, as indicated in column 1 of Table 6).

Our pattern extraction algorithm has two distinct parts. First, the input forms are left-aligned, as indicated in column 2 of Table 6. Second, all vertically identical characters are replaced by a placeholder, merging contiguous placeholders, as indicated in column 3 of Table 6. This allows us to discard constant information, and keep only the information that varies and their position in the form. We then group the resulting strings two by two to form the patterns, as indicated in column 4.

To model the local strategy, we proceed in exactly the same fashion, except for the fact that the algorithm is applied separately to each

Table 6: Plural present forms for the verb AMENER 'bring': Global pattern extraction

	1. Input	2. Left aligned forms				3. Variables
PRS.1PL	amønõ	am	ø	n	õ	…ø…õ
PRS.2PL	amøne	am	ø	n	e	…ɛ…
PRS.3PL	amɛn	am	ɛ	n		…ø…e

4. Output: patterns	
PRS.1PL ~ PRS.2PL	…ø…õ ~ …ø…e
PRS.2PL ~ PRS.3PL	…ø…e ~ …ɛ…
PRS.3PL ~ PRS.1PL	…ɛ… ~ …ø…õ

	1. Input	2. Left aligned forms					3. Variables
PRS.1PL	amønõ	a	m	ø	n	õ	…õ
PRS.2PL	amøne	a	m	ø	n	e	…e
PRS.2PL	amøne	a	m	ø	n	e	…ø…e
PRS.3PL	amɛn	a	m	ɛ	n		…ɛ…
PRS.3PL	amɛn	a	m	ɛ	n		…ɛ…
PRS.1PL	amønõ	a	m	ø	n	õ	…ø…õ

Table 7: Plural present forms for the verb AMENER 'bring': Local pattern extraction

4. Output: patterns	
PRS.1PL ~ PRS.2PL	…õ ~ …e
PRS.2PL ~ PRS.3PL	…ø…e ~ …ɛ…
PRS.3PL ~ PRS.1PL	…ɛ… ~ …ø…õ

pair of paradigm cells, rather than just once to the whole set of pairs. In the case at hand, as indicated in Table 7, this leads to three separate runs of the algorithm, leading in each case to the production of one pattern.

As we see from the tables, the local strategy produces binary alternation patterns which encode strictly local knowledge about the pair, while global alternation patterns encode knowledge about the rest of the paradigm. On this small paradigm, the choice of strategy only makes a difference for the alternation between the first and second person. The global strategy yields a pattern specific to verbs with an /ə/ in the penultimate syllable. The local strategy, on the other hand, yields a more general pattern, that also characterises verbs with no /ə/ in the penultimate syllable. This is relevant to clustering, as the global strategy, but not the local strategy, will take AMENER to exhibit a rather unusual behavior.[7]

Both strategies take the surface forms at face value and do not attempt to derive any underlying representations. Alternations are thus morpho-phonological rather than strictly morphological. There are two main reasons for this choice: First, it is not clear how to automatically abstract all regular phonology from a set of wordforms (our

[7] In fact, all French verbs except ÊTRE 'be', FAIRE 'do', DIRE 'say' and their derivative use the same pattern as AMENER.

input). Second, some regular phonological alternations do contribute to opacities in alternations, and are predictible only in one direction. Abstracting them out would be to underestimate the task speakers face when they inflect forms.

As this example illustrates, our current algorithm is able to capture stem-internal alternations that are rampant in familiar inflection systems. Actually, it is general enough to allow for multiple points of variation within the string, and hence is in principle capable of dealing with root-and-pattern morphology. On the other hand, the use of left-alignment is a clear limitation of the algorithm, making it impossible to capture systems making any use of prefixation.[8] While this is a clear limitation, it has no influence on performance on non-prefixing systems such as the ones we will explore in Section 4.

2.2 *From patterns to classes*

2.2.1 Previous work on inflection classes inference

The task of automatically inferring inflection classes has recently seen growing interest.

An early attempt at that task by Goldsmith and O'Brien (2006) used a neural network to relate features to exponents. The hope was that the hidden layer of the network would reflect inflectional classification. However, experiments on both Spanish and German failed to produce such a result. Very recently, Malouf (2017) has developed more promising uses of neural networks to model inflectional behavior, but the results cannot be interpreted straightforwardly as a partition of inflectional macroclasses.

There have also been efforts in NLP to infer microclasses from incomplete paradigms (Eskander *et al.* 2013; Monson *et al.* 2004), building on the same kinds of methods used by Dreyer and Eisner (2011) and Durrett and DeNero (2013); Nicolai *et al.* (2015) for inflectional realisation in sparse lexica.

More directly related to the present work is Brown and Evans (2012), who present an attempt at infering inflection classes for the system of Russian nouns. They evaluate redundancy between

[8] See Beniamine (2017) for a pattern inference algorithm capturing prefixation, suffixation, infixation, root-and-pattern morphology, and suprasegmental exponence, that could readily be used as a substitute for the simple algorithm used in this paper.

paradigms through a compression distance. They perform clustering on this basis using CompLearn (Cilibrasi and Vitanyi 2005). The output of CompLearn is an unrooted binary tree. Since this tree is hardly interpretable, Brown and Evans use a series of heuristics to select preferred nodes in the tree. Their approach does not rely on the abstraction of inflectional realisations. Since the compression distance is computed on forms, it captures as much, if not more, of the similarity between stems than the similarity of the inflectional material.It is then unclear whether the resulting tree encodes strictly inflectional structure. Since Brown and Evans (2012)'s goal is to validate an account of Russian noun inflection (Brown 1998), they are attempting to decide which heuristic yields an inflectional classification that is presupposed to be correct. If we do not rely on a pre-existing theory, we also lose the way to choose among such heuristics. In this paper, we thus wish to infer a partition of classes directly.

Bonami (2014) attempts to improve on Brown and Evans's (2012) strategy by inferring inflectional realisations as a separate step. He produces inflectional classification trees based on both affixes and alternation patterns, which corresponded broadly to our local and global segmentation strategies. The trees are built using distance-based agglomerative hierarchical clustering with average linkage (Sokal and Michener 1958). Unfortunately, the distances used for the alternation patterns and for the exponents are not commensurable. Moreover, the final shape of the inflectional system is a tree with no distinguished macroclass level. Indeed, since distances evaluate the fitness of one class, not the fitness of a partition, they are not an appropriate tool with which to choose a preferred partition of classes in the tree.

Lee and Goldsmith's (2013) approach is closest to ours. Starting from a representation of paradigms, they define a greedy clustering algorithm that uses the Minimal Description Length principle (Rissanen 1978) to decide which paradigms it is optimal to group together in a cell of the partition. Note that this is closely related to the use of MDL to compare inflection class systems (Sagot and Walther 2011; Walther and Sagot 2011; Walther 2013), for which see Section 1.3.5, but improves on it by using MDL as a criterion for clustering rather than using it to compare manually crafted classifications. However, Lee and Goldsmith's approach is marred by what we take as a poor choice of representation for paradigms. In their approach, paradigms

are collections of words, and words are represented by the set of characters in their orthographic forms. For instance, *delay* and *delayed* are represented by the same set {a,d,e,l,y}. This is unsatisfactory in many respects: such representations lack any plausibility as representations of the knowledge of speakers, and make it impossible to take into account important aspects of morphological structure. For instance, the character sets of *daring* ({a,d,g,i,n,r}) is closer to that of *denigrate* ({a,d,e,g,i,n,r,t}) than to that of *dare* ({a,d,e,r}).

The approach presented below can be seen as an attempt to combine ideas from Bonami on the use of alternation patterns to assess similarity between lexemes, and from Sagot and Walther and from Lee and Goldsmith on the use of the Minimal Description Principle as a criterion.

2.2.2 Our approach to inflection classes inference

Our goal is to infer a partition of macroclasses on top of microclasses directly. Doing so requires formal definitions of both of these constructs. We take microclasses to follow the strictest definition of inflection classes:

(5) A system of *microclasses* is a partition of the set of lexemes into classes which share the exact same list of inflectional realisations.

It follows that the microclasses can be transparently deduced from the inflectional realisation. We propose to define macroclasses as follows:

(6) A system of *macroclasses* is an optimal system of non-overlapping sets of microclasses.

To decide which partition is optimal, we now need a criterion to compare different partitions of a set of microclasses.

The leading idea is to look for the system of macroclasses that optimally captures the regularities in the data. Let's say we begin with a system of microclasses and wish to merge some of them into broader macroclasses. In the initial system, each microclass is described separately as having a list of patterns indexed by pairs of cells. Wherever merged microclasses have a common pattern, an optimal description will be able to mention that pattern only once by associating it with the

merged class. On the other hand, if merged classes use distinct patterns for the same cell, any description will need to disambiguate which microclass uses which pattern. Following Occam's razor, merging microclasses into a macroclass can then be seen as beneficial to concision as long as we gain more due to common patterns than we lose because of disambiguation. This follows the overall intuition of the Minimal Description Length Principle, according to which the structure best fitting a dataset is the structure allowing for the shortest description of the data. However, the reason we choose that structure is not that concision is a quality *per se*, but rather that it reflects the ability of the structure to account for regularities in the data. Thus, we decide that a partition of the set of lexemes in macroclasses is better than another one if it leads to a more concise description of the inflection class system.

In the next section, we present the probabilistic model that allows us to assess the length of a description, and the algorithm that makes use of this criterion to find the best macroclasses for a given set of microclasses.

3 FINDING AN OPTIMAL PARTITION

3.1 *The minimum description length principle*

Minimum Description Length (MDL) is a general framework for selecting an appropriate model of a dataset within a space of possible models (Rissanen 1984; Grünwald 2007). The underlying idea is that wherever there is structure in a dataset, that structure can be used to provide a shorter description of the dataset. Different models will capture the structure in the data to different extents. The quality of a model can thus be assessed by looking at the length of an optimal description of the data relying on the model. This will comprise both the description of the model itself, and a description of whatever aspects of the data the model was not able to describe. Optimality of the description is ensured in information-theoretic terms. The Minimal Description Length Principle then states that the best model is the model leading to the shortest description. This is supposed to embody Occam's razor: the best model is the most frugal model. For the MDL principle to make sense, it is essential that the models under consideration be strictly commensurable. MDL allows one to compare

different models written in the same formal framework, not all conceivable models, an endeavour that has been proved mathematically to be impossible.

The MDL is a general method for inductive inference, used mostly in the field of machine learning as a sound way of avoiding overfitting. In recent years, it has been used to address problems of linguistic modeling in morphology in two very different ways. As mentioned above, Sagot and Walther (2011, 2013) and Walther (2013) compare hand-designed descriptions of the same inflection system couched in the same rich formalism and use description length to decide which of these is preferable. Goldsmith (2001) then again explores automatically all possible morphological segmentations of a text (hence using a coarse-grained formalism for morphological description) and uses description length of the whole text to decide which segmentation is more likely to be correct.

In this paper, we adopt from Sagot and Walther the idea of using a description-length-based information-theoretic criterion for comparing competing accounts of a morphological system. However, we make use of this idea in a different setting; their approach, as Goldmsith's approach, is constructive in the sense of Blevins (2006); They are looking for the shortest possible grammar that generates the data within a predefined framework. This contrasts with the work reported in this paper, where we compare descriptions that are highly redundant. We make no claim that these descriptions are reasonable. We only claim that comparing them is useful to assess which set of macroclasses best represents regularities and irregularities in the data. Although this may be less familiar to linguists, this is actually the standard use of MDL in statistical inference, where descriptions are constructed for the purposes of comparing models, and do not necessarily have an inherent value.

3.2 *Modeling macroclass systems*

For the purposes of comparing inflection class systems, we thus need to define formally a family of models of inflection systems that differ in the way they group lexemes in classes, and then to assess their description length. The shape of the models we will use follows from the view of the inflectional macroclasses we argued for above. Lexemes are grouped in microclasses according to which patterns they instan-

tiate, a microclass being a class of lexemes that instantiate the exact same vector of patterns; macroclasses form a partition of the set of microclasses. A model of an inflection class system will contain the following four components:

(7) a. A specification M of which lexemes belong to which microclasses.

b. A specification C of which microclass belongs to which macroclass or CLUSTER of microclasses.

c. A specification $\mathscr{P}$ of which patterns (for each pair of paradigm cells) are instantiated in each cluster. Note that for any cluster containing more than one microclass, there will be at least one pair of cells for which two or more patterns are instantiated; otherwise there would only be one inflectional behavior and hence only one microclass in the cluster.

d. The residual information R that cannot be deduced from the assignment of a microclass to a cluster. This amounts to specifying, wherever a cluster instantiates more than one pattern for a pair of cells, which microclass in the cluster uses which pattern.

To better understand how such models can be used to compare candidate systems of macroclasses, let us consider a toy system consisting of the three French verbs AMENER 'bring', BOIRE 'drink' and DIRE 'say' in the indicative present plural. Table 8 indicates both the raw (sub)paradigms of the three verbs and the patterns abstracted from these paradigms under a local pattern inference strategy. The three verbs clearly belong to three different microclasses. Let us consider then in turn the three possible ways of grouping them into macroclasses. Table 9 provides an informal but rather detailed specification of the four components of a description of three possible classifications of this dataset. In each case, two of the three verbs are grouped together in a cluster, and the remaining third verb forms a cluster of its own.

As should be apparent from the table, the three candidate classifications do not differ in the length of a description of the assignments of lexemes to microclasses or microclasses to clusters. However they

Table 8: Subparadigms and local patterns for three French verbs in the Indicative Present Plural

	Raw data			Patterns (local strategy)		
	1PL	2PL	3PL	1PL∼2PL	1PL∼3PL	2PL∼3PL
AMENER	amənɔ̃	amәne	amɛn	…ɔ̃∼e (p_1)	…ə…ɔ̃∼…ɛ… (p_3)	…ə…e∼…ɛ… (p_6)
BOIRE	byvɔ̃	byve	bwav	…ɔ̃∼e (p_1)	…y…ɔ̃∼…wa… (p_4)	…y…e∼…wa… (p_7)
DIRE	dizɔ̃	dit	diz	…zɔ̃∼…t (p_2)	…ɔ̃∼… (p_5)	…t∼…z (p_8)

Table 9: Detailed description of three classifications of the paradigms from Table 8 in microclasses and macroclasses

Partition	{{AMENER},{BOIRE,DIRE}}	{{AMENER, BOIRE},{DIRE}}	{{AMENER,DIRE},{BOIRE}}
M	AMENER $\mapsto m_1$ BOIRE $\mapsto m_2$ DIRE $\mapsto m_3$	AMENER $\mapsto m_1$ BOIRE $\mapsto m_2$ DIRE $\mapsto m_3$	AMENER $\mapsto m_1$ BOIRE $\mapsto m_2$ DIRE $\mapsto m_3$
C	$m_1 \mapsto c_1$ $m_2 \mapsto c_2$ $m_3 \mapsto c_2$	$m_1 \mapsto c_1$ $m_2 \mapsto c_1$ $m_3 \mapsto c_2$	$m_1 \mapsto c_1$ $m_2 \mapsto c_2$ $m_2 \mapsto c_1$
$\mathscr{P}$	c_1 : 1PL ∼ 2PL : $\{p_1\}$ 1PL ∼ 3PL : $\{p_3\}$ 2PL ∼ 3PL : $\{p_6\}$ c_2 : 1PL ∼ 2PL : $\{p_1, p_2\}$ 1PL ∼ 3PL : $\{p_4, p_5\}$ 2PL ∼ 3PL : $\{p_7, p_8\}$	c_1 : 1PL ∼ 2PL : $\{p_1\}$ 1PL ∼ 3PL : $\{p_3, p_4\}$ 2PL ∼ 3PL : $\{p_6, p_7\}$ c_2 : 1PL ∼ 2PL : $\{p_2\}$ 1PL ∼ 3PL : $\{p_5\}$ 2PL ∼ 3PL : $\{p_8\}$	c_1 : 1PL ∼ 2PL : $\{p_1, p_2\}$ 1PL ∼ 3PL : $\{p_3, p_5\}$ 2PL ∼ 3PL : $\{p_6, p_8\}$ c_2 : 1PL ∼ 2PL : $\{p_1\}$ 1PL ∼ 3PL : $\{p_4\}$ 2PL ∼ 3PL : $\{p_7\}$
R	$m_2 : p_1$ $m_3 : p_2$ $m_2 : p_4$ $m_3 : p_5$ $m_2 : p_7$ $m_3 : p_8$	$m_1 : p_3$ $m_2 : p_4$ $m_1 : p_6$ $m_2 : p_7$	$m_1 : p_1$ $m_3 : p_2$ $m_1 : p_3$ $m_3 : p_5$ $m_1 : p_6$ $m_3 : p_8$

differ both in terms of assignment of patterns to clusters and in terms of residual information: because the second classification groups together two microclasses that share a pattern, the assignment of patterns to clusters is briefer (pattern p_1 is only mentioned once rather than twice), as is the residue (the clusters provide perfectly accurate information on 1PL ∼ 2PL, and hence the residue makes no mention of patterns p_1 and p_2). Hence the second classification, grouping together AMENER and BOIRE, leads to a shorter description and should be preferred over the other two.

Two more classifications have to be considered: a classification with only one macroclass, and one with one macroclass per microclass. These are illustrated in Table 10. In the first case, all of the

Table 10: Detailed description of two extreme classifications of the paradigms from Table 8 in microclasses and macroclasses

Partition	{{AMENER, BOIRE,DIRE}}	{{AMENER},{DIRE},{BOIRE}}
M (microclasses)	AMENER ↦ m_1 BOIRE ↦ m_2 DIRE ↦ m_3	AMENER ↦ m_1 BOIRE ↦ m_2 DIRE ↦ m_3
C (macroclasses)	$m_1 \mapsto c_1$ $m_2 \mapsto c_1$ $m_3 \mapsto c_1$	$m_1 \mapsto c_1$ $m_2 \mapsto c_2$ $m_3 \mapsto c_3$
$\mathscr{P}$ (patterns)	c_1 : 1PL ∼ 2PL : $\{p_1, p_2\}$ 1PL ∼ 3PL : $\{p_3, p_4, p_5\}$ 2PL ∼ 3PL : $\{p_6, p_7, p_8\}$	c_1 : 1PL ∼ 2PL : $\{p_1\}$ 1PL ∼ 3PL : $\{p_3\}$ 2PL ∼ 3PL : $\{p_6\}$ c_2 : 1PL ∼ 2PL : $\{p_1\}$ 1PL ∼ 3PL : $\{p_4\}$ 2PL ∼ 3PL : $\{p_7\}$ c_2 : 1PL ∼ 2PL : $\{p_2\}$ 1PL ∼ 3PL : $\{p_5\}$ 2PL ∼ 3PL : $\{p_8\}$
R (residue)	$m_1 : p_1$ $m_2 : p_1$ $m_3 : p_2$ $m_1 : p_3$ $m_2 : p_4$ $m_3 : p_5$ $m_1 : p_6$ $m_2 : p_7$ $m_3 : p_8$	

disambiguation is done in the residue, while in the second the same thing is done in the pattern assignment. The table gives the impression that the first description is longer, as it has both something in $\mathscr{P}$ and in R. However, it actually captures a generalisation that the other does not. In information-theoretic terms, it is a shorter description.

Going from this informal presentation to a precise measure of description length requires one to provide an explicit scheme for describing each of M, C, $\mathscr{P}$ and R as sequences of symbols. Any such sequence displays a probability distribution of the symbols via their relative frequency in the message. Information Theory (Shannon 1948) provides a way of determining the size in bits of the shortest possible encoding of that message.

Intuitively, this depends on the length of the message (all other things being equal, longer messages are longer to encode), and the frequency of the symbols within the message (symbols that occur multiple times in the message are less surprising and hence less costly). More precisely, the length of the shortest possible description of a message m is the length of message times the entropy of the distribution of the list S of symbols in the message.

$$\begin{aligned}(8)\qquad \mathrm{DL}(m) &= |m| \cdot H(m) \\ &= -|m| \cdot \textstyle\sum_{x \in S} P(x) \cdot \log_2 P(x) \\ &= -\textstyle\sum_{x \in S} \mathrm{count}(x) \cdot \log_2 \frac{\mathrm{count}(x)}{|m|}\end{aligned}$$

The appendix presents in detail the scheme we used in this paper. For present purposes, it is sufficient to note that we define the description length of an inflection system to be the sum of the description lengths of its four components.

$$(9)\qquad \mathrm{DL}(I) = \mathrm{DL}(M) + \mathrm{DL}(C) + \mathrm{DL}(\mathscr{P}) + \mathrm{DL}(R)$$

3.3 *Searching for possible partitions*

We can now define our criterion for deciding which of a set of partitions is optimal as minimisation of DL(I). Therefore, searching for the macroclasses could theoretically be a matter of evaluating all the possible partitions over the microclasses. This is not a realistic strategy in practice. For a system with 15 microclasses, there are more than a billion different partitions to consider. For a system such as French conjugation, with 74 microclasses, the number of partitions to consider

approaches the number of atoms in the universe (10^{80}).[9] The size of the search space entails that a full exploration of all possibilities is out of the picture. Here we use a greedy bottom-up search, which finds macroclasses from microclasses by merging repeatedly two clusters.

The algorithm can be described as follows:

(10) a. Start with a partition where each microclass is a cluster.
b. For each pair of clusters, evaluate what the DL of the system would be if the pair were to be merged.
c. Merge one of the pairs of clusters which results in a minimal DL.
d. Repeat steps (b-c) until the DL stops decreasing.

We exemplify the search with an imaginary system of five microclasses, named from A to E. Figure 1 illustrates how the algorithm proceeds. The numbers used here as description lengths are arbitrary and serve only the purpose of illustrating the algorithm.

Step (1) corresponds to the initial state, where each microclass forms its own cluster. Let us assume arbitrarily that the description length of the corresponding model is of 6 bits. In step (2), we select the pair of microclasses leading to the lowest DL. That is, we examine the 10 models obtained by putting any two microclasses in the same clusters, and pick the one whose description length is the smallest. In this instance it happens to be D and E, with a DL of 4.

We then proceed to determine again the optimal merges for the system constituting the output of step (2). In this instance, it happens that there are two optimal solutions: merging A and B or A and C both leads to models with a description length of 3.5. In such a situation, we choose one of the optimal solutions at random. Here the choice happens to be merging A and C.

[9] The number of possible partitions for a set of cardinality n is the n^{th} Bell number B_n, where $B_0 = 1$ and:

$$B_{n+1} = \sum_{k=0}^{n} \binom{n}{k} B_n.$$

The Bell numbers grow very quickly—much more quickly than an exponential function, for instance.

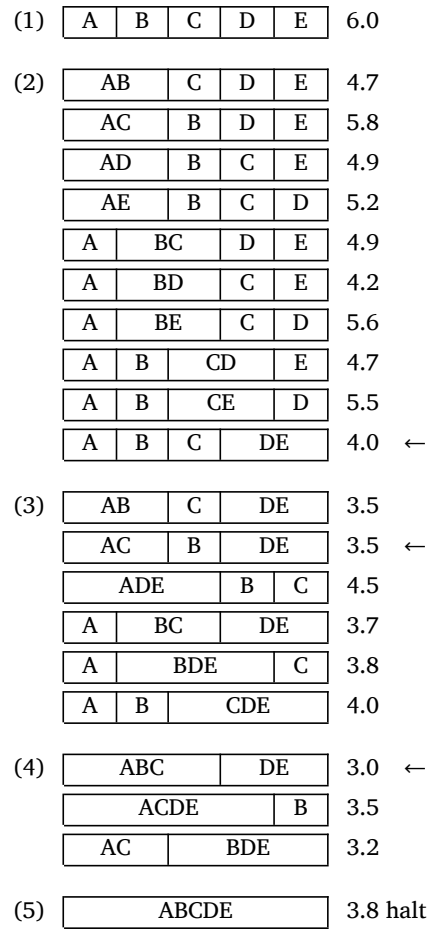


Figure 1:
Example of a run of the search algorithm

In step (4), we examine all possible merges and find that only one merge, ACB, leads to an optimal model. Finally, in step (5), we examine the result of merging the two only remaining macroclasses in a single cluster. This however leads to a description length that is longer than that of the optimal description found at step (4). This shows that merging clusters has stopped being beneficial for description length, and we conclude that the partition found at the end of step (4) is optimal.

Three important remarks about the algorithm are in order. First, there is no *a priori* guarantee that there will be several macroclasses. It is possible, if the DL continues to lower, to end up with only one

cluster. Thus this algorithm is suited to decide on an empirical basis if a system displays non-trivial macroclasses. Second, our algorithm is nondeterministic: at step (3) in the example above, we had to choose at random which classes to merge, which entails that a different choice might have taken us to a different final partition in two macroclasses. To address this issue, in the empirical studies below, we will perform multiple runs of the algorithm and check that the results are stable. Third, as with most greedy algorithms, we can only hope that the local optimum found by the algorithm indeed corresponds to the global optimum—and hence that the macroclasses we find are indeed the true macroclasses. While this is not fully satisfactory, we know of no search algorithm able to find the global optimum in a reasonable amount of time.

4 CLASSIFICATION AND RESULTS

In this section we discuss the results of applications of our algorithms to the conjugation of French and European Portuguese, and address three research questions: first, as we saw in the last section, not all datasets will lead to the emergence of a partition into macroclasses; the algorithm may terminate with the conclusion that introducing macroclasses does not lead to a more economical description. Given this, do macroclasses emerge in the systems at hand? Second, we introduced in section 2 two ways of describing inflectional realisations, relying on either a local or a global strategy. Where they emerge, how different are the macroclass systems found with both strategies? Third, how do the macroclass systems inferred by our algorithm compare to the systems posited by descriptive morphologists?

4.1 *Datasets*

Our datasets take the form of large inflectional lexica with phonemically transcribed forms.

For French, we rely on the verbal subset of the Flexique dataset (Bonami *et al.* 2014). It is based on the Lexique dataset (New *et al.* 2001), but the transcriptions have been corrected by hand, and the incomplete paradigms provided by Lexique have been filled semi-automatically. We ignore any defective or overabundant entries. The resulting dataset contains 5259 lexemes each containing forms for each of 51 morphosyntactic cells.

For European Portuguese, we rely on the European Portuguese pronunciation dictionary elaborated by Veiga *et al.* (2013), kindly provided and adapted by hand by Fernando Perdigão. The dataset contains 1995 lexemes, with forms for each of the 69 morphosyntactic cells. As was done for French, overabundance and defectivity are ignored.

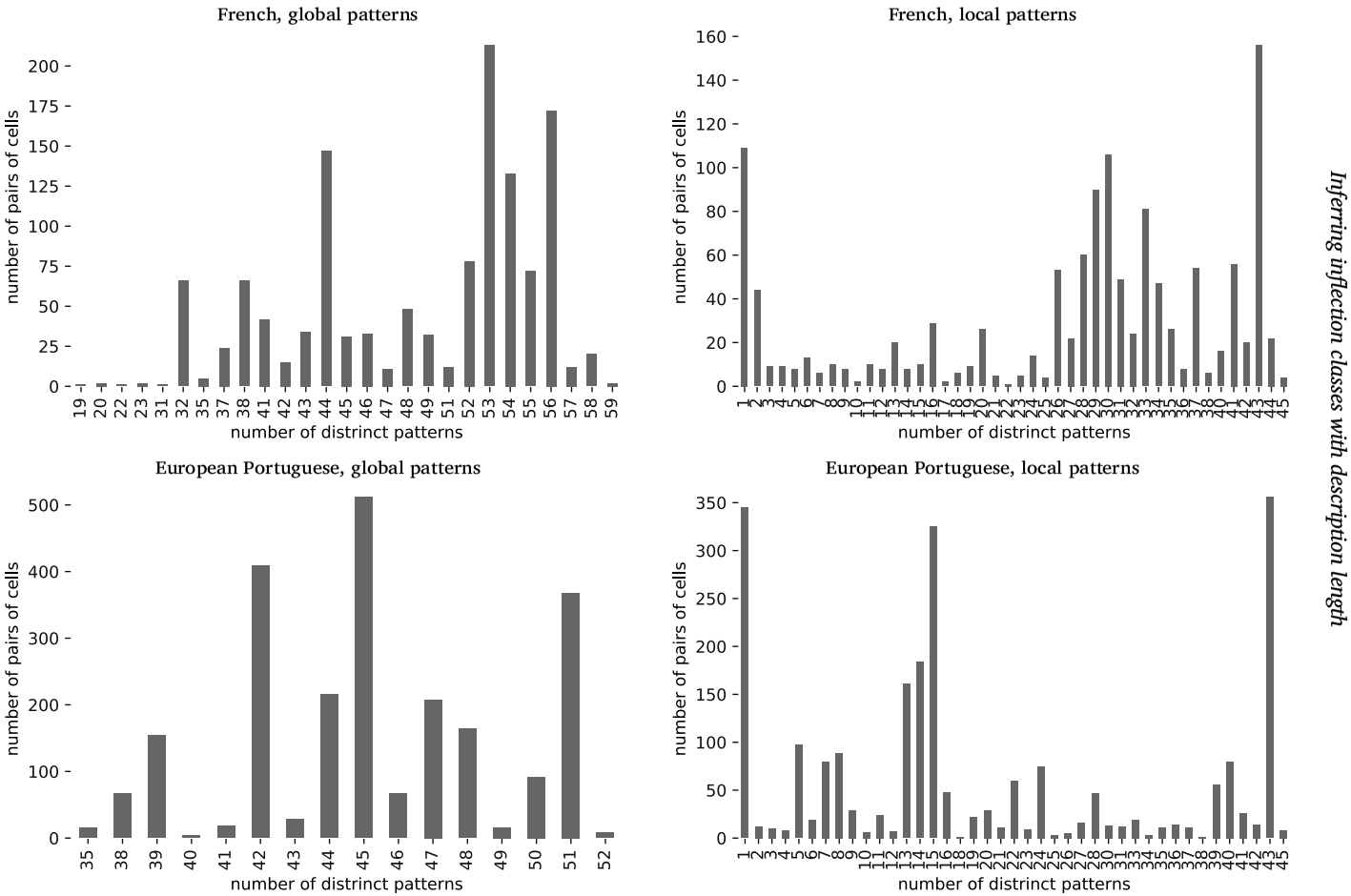
To compute phonological generalisations for on the context in which patterns are satisfied, our program also requires as input a specification of the value of each character as a vector of phonological features. We used the feature descriptions designed for the purposes of Bonami and Boyé (2014) and Bonami and Luís (2014).[10]

For both datasets, we ran the algorithm twice: once with alternation patterns found using the local strategy, and once with those found using the global strategy. The result consists of two classifications: first, that of lexemes into microclasses, then the classification of microclasses into macroclasses. The program also logs the history of the classification process as a tree of successive merges.

4.2 *Patterns*

The local and global alternation patterns differ substantially. As could be expected, the global approach results in a larger number of patterns per pair of cells, as is shown in Figure 2. This is due to the fact that any irregularity in the relation between two cells will have an impact on which patterns relate all other pairs of cells of that lexeme. For example, in a situation where two paradigm cells are identical for all lexemes, the local approach finds that generalisation, while the global approach may find more than one pattern depending on what happens elsewhere.

[10] One notable choice for the French dataset is that height distinctions between mid-vowels were neutralised by using the same feature matrices for the pairs of vowels ([e],[ɛ]), ([ø],[œ]), and ([o],[ɔ]). This is motivated by the fact that mid-vowel pronunciations are in a state of fluctuation in standard French in some positions, so that in some cases no single narrow transcription is appropriate for a given word. In examples below the neutralised vowels are noted E, Ø and O respectively.



These figures read as follows: In the French global setting, there are 213 pairs of cells which contain 53 distinct patterns, 172 pairs of cells which contain 56 distinct patterns, etc.

Figure 2: Distribution of the number of distinct patterns per pair of cells in each setting

4.3 *Microclasses*

Remember that microclasses are sets of lexemes exhibiting identical patterns for all pairs of cells. Even though the two strategies find very different patterns, in both languages, they lead to the same inventory of microclasses. This is a general property of the algorithm that is best explained by observing that two lexemes show an identical global inflectional behavior if and only if they show an identical behavior in each local context.

For French, we find 73 microclasses. The largest class contains verbs with the same inflectional behavior as AXER (3671), followed by the class of verbs behaving like AGIR (353). 60 of the classes have less than 20 members, with 15 having just one member. For European Portuguese, we find 55 microclasses, the largest of which contains verbs behaving like USAR (911), followed by that of verbs such as JOGAR (177). 43 microclasses present less than 20 members, with 15 having just one member.

Microclasses have little value as generalisations over inflectional behavior, because any small deviation between the behavior of two lexemes results in separate classes.

4.4 *Macroclasses*

Since microclasses with local and global patterns display different similarity structures, they also produce different macroclass systems. We ran the macroclass algorithm over the four different microclass systems (French and European Portuguese, local and global). The history of the algorithm can be depicted as a tree of recursive merges. Figures 3, 4, 5 and 6 show the history for both the local and global patterns. Black arcs represent merges where the description length decreased, gray arcs merges where the description length did not decrease; hence macroclasses are those clusters dominating black arcs and dominated by gray arcs. Nodes corresponding to a macroclass are labelled with the number of lexemes in the class.

We observe that with the global strategy, most microclasses do not cluster much together, while the local strategy leads to fewer macroclasses that seem more balanced. It is important to note here that the intermediate merges cannot be given a straightforward interpretation: their order does not necessarily reflect anything relevant,

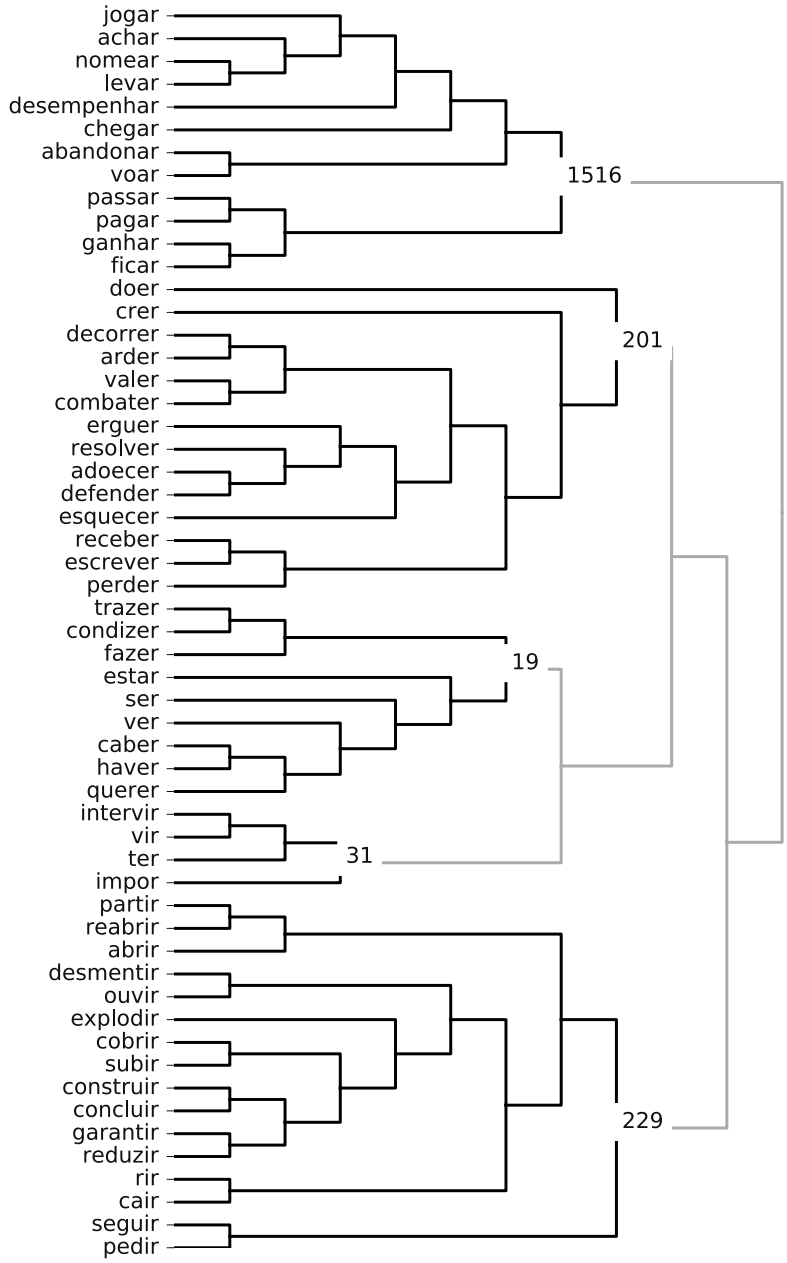


Figure 3: History of merges for European Portuguese macroclasses, on local patterns

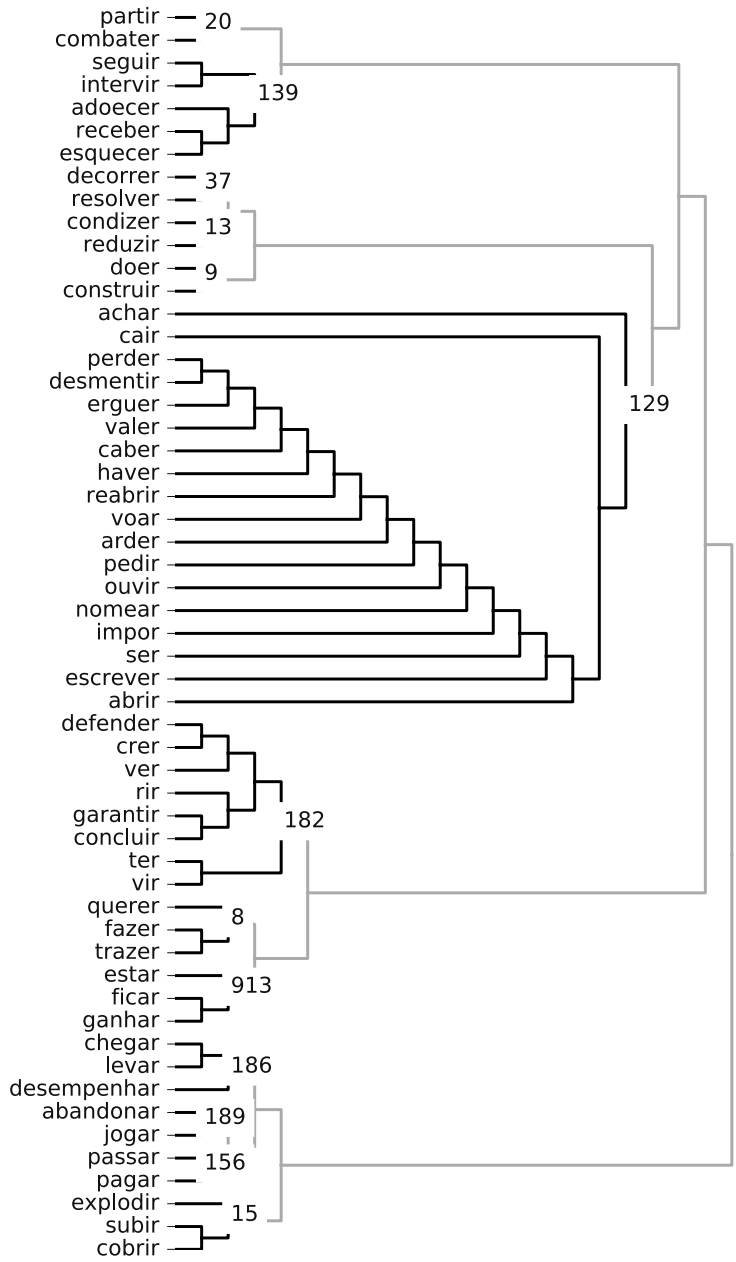


Figure 4: History of merges for European Portuguese macroclasses, on and global pattern.

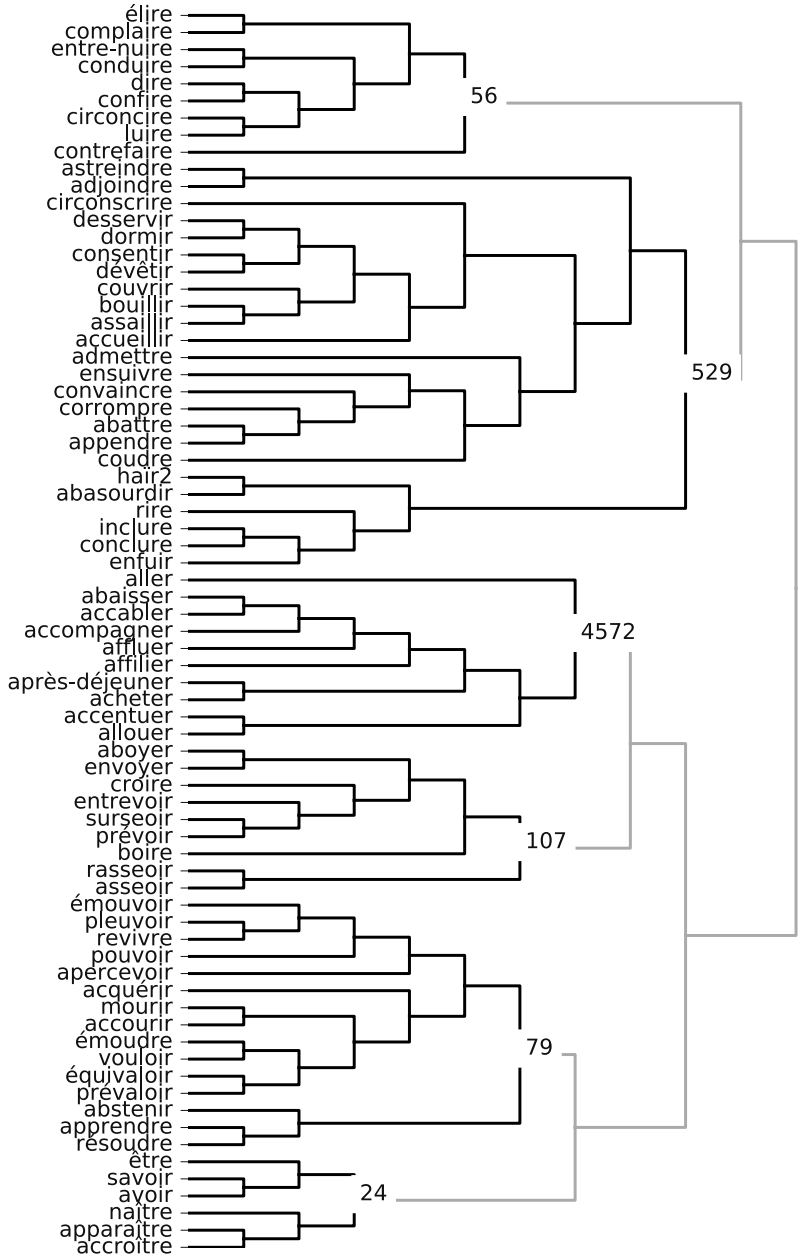


Figure 5: History of merges for European Portuguese macroclasses, on local patterns

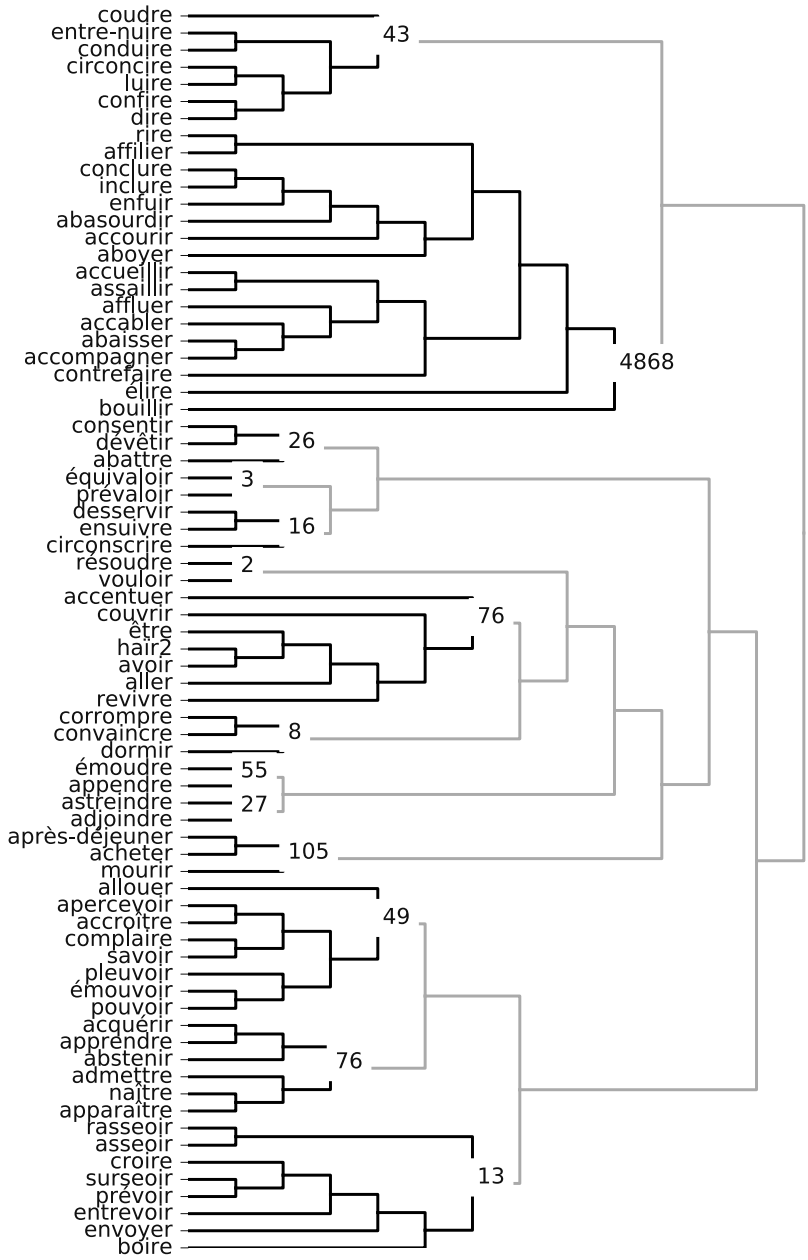


Figure 6: History of merges for European Portuguese macroclasses, on global patterns

and there is little reason to believe that they represent classes of intermediate granularity.

Remember that the greedy algorithm which we used to merge classes is nondeterministic: if we happen to encounter two competing best merges leading to the same decrease in DL, the algorithm chooses at random which to perform. To ensure the stability of our results despite this non-determinism, for each condition, we ran the classification procedure 100 times. The order of merges varied, especially at the beginning of runs, but the macroclass partition was constant over iterations. Figure 7 represents the intersection of 100 history trees for the French local patterns condition: if we consider each node as represented by the set of leaves it spans, and each edge as a pair of nodes, this history tree keeps only nodes and edges common to all 100 iterations, then adds edges (dashed in the figure) according to node spans to keep a tree structure. As can be seen on the picture, the areas of variation are small and localised at the bottom of the tree (the start of the algorithm). Results in the three other settings are similar.

In all settings, we do find non-trivial macroclasses: the clustering process stops before having merged all microclasses together. For European Portuguese verbs, we find 13 macroclasses with the global patterns and 5 with the local patterns. For French verbs, we find 14 macroclasses with the global patterns and 6 with the local patterns.

In neither condition did we find a bipartition between microclasses usually deemed regular and those that are usually deemed irregular. This suggests that a classification based on regularity and a classification based on similarity will be orthogonal to one another.

In both languages, the global strategy leads to classifications that contain numerous small macroclasses and bear no resemblance to extant classifications for these languages. Local patterns lead to fewer macroclasses, and generalisations are highly similar to traditional wisdom. This is clearly due to local patterns capturing more fine-grained similarity. We take this to suggest that our algorithm, applied under a local strategy to pattern inference, is close to operationalizing the heuristics used by descriptive linguists when designing a hand-made classification.

In French, the grammatical tradition distinguishes three conjugations. The first conjugation consists of verbs with infinitives in *-er*. The second conjugation consists of verbs with infinitives in *-ir* and exhibit-

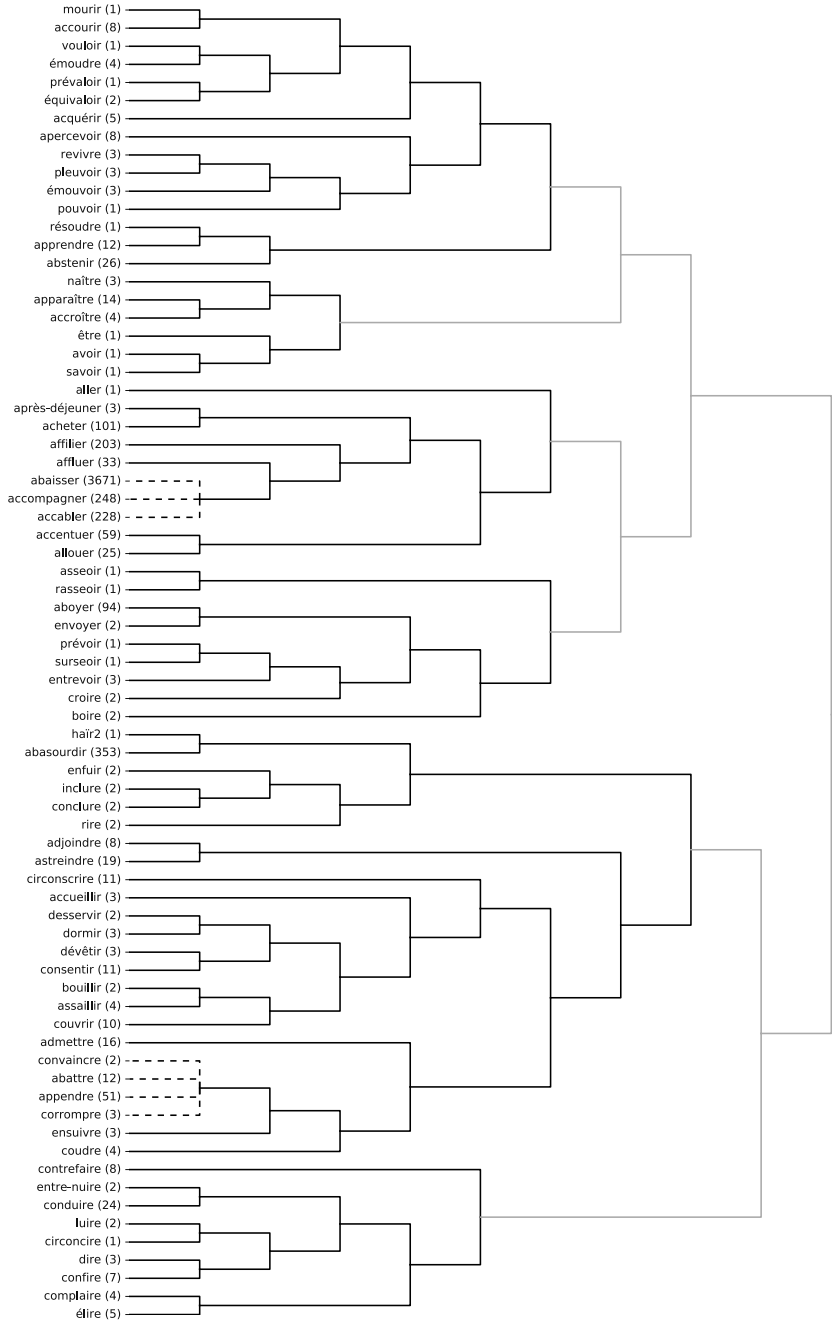


Figure 7: Intersection of the history of 100 runs for the French local patterns condition

Macroclass 1	résoudre (1), vouloir (1)
Macroclass 2	adjoindre (8), astreindre (19)
Macroclass 3	circoncire (1), conduire (24), confire (7), coudre (4), dire (3), entre-nuire (2), luire (2)
Macroclass 4	appendre (51), émoudre (4)
Macroclass 5	asseoir (1), boire (2), croire (2), entrevoir (3), envoyer (2), prévoir (1), rasseoir (1), surseoir (1)
Macroclass 6	convaincre (2), corrompre (3), dormir (3)
Macroclass 7	abstenir (26), acquérir (5), admettre (16), apparaître (14), apprendre (12), naître (3)
Macroclass 8	abaisser (3671), abasourdir (353), aboyer (94), accabler (228), accompagner (248), accourir (8), accueillir (3), affilier (203), affluer (33), assaillir (4), bouillir (2), conclure (2), contrefaire (8), enfuir (2), inclure (2), rire (2), élire (5)
Macroclass 9	accentuer (59), aller (1), avoir (1), couvrir (10), haïr2 (1), revivre (3), être (1)
Macroclass 10	accroître (4), allouer (25), apercevoir (8), complaire (4), pleuvoir (3), pouvoir (1), savoir (1), émouvoir (3)
Macroclass 11	abattre (12), consentir (11), dévêtir (3)
Macroclass 12	acheter (101), après-déjeuner (3), mourir (1)
Macroclass 13	prévaloir (1), équivaloir (2)
Macroclass 14	circonscrire (11), desservir (2), ensuivre (3)

Table 11: French macroclasses to microclass mapping (global strategy)

ing an *-i-/-iss-* stem alternation, while the third conjugation consists of all remaining verbs. Remember that Kilani-Schoch and Dressler (2005) take irregularity as a criterion in grouping the traditional second and third conjugations. See also Plénat (1987) for arguments to the effect that the second and third conjugation pattern together, at least in the formation of the simple past and past participle.

The simulations we ran both show that the traditional third conjugation is very heterogeneous, as its members always end up in different macroclasses. The global approach does not seem to capture the intuition of macroclass that has been described by linguists, showing 14 macroclasses, some of which contain a very small number of lexemes (Table 11), and none of which resembles by any stretch a traditional conjugation.

In contrast, the local strategy leads to a classification that is mostly congruent with the traditional approach (Table 12).

All verbs from the traditional first conjugation are clustered together, except ABOYER and ENVOYER, which indeed exhibit alterna-

Table 12: French: Comparison of inferred macroclasses (on local patterns) vs traditional conjugations

Macroclasses	Traditional	Lexemes
Macroclass 1	3rd conj.	circoncire (1), complaire (4), conduire (24), confire (7), contrefaire (8), dire (3), entrenuire (2), luire (2), élire (5)
Macroclass 2	3rd conj.	abstenir (26), accourir (8), acquérir (5), apercevoir (8), apprendre (12), mourir (1), pleuvoir (3), pouvoir (1), prévaloir (1), revivre (3), résoudre (1), vouloir (1), émoudre (4), émouvoir (3), équivaloir (2)
Macroclass 3	first conj.	abaisser (3671), accabler (228), accentuer (59), accompagner (248), acheter (101), affilier (203), affluer (33), allouer (25), après-déjeuner (3)
	3rd conj.	aller (1)
Macroclass 4	second conj.	abasourdir (353), haïr2 (1)
	3rd conj.	abattre (12), accueillir (3), adjoindre (8), admettre (16), appendre (51), assaillir (4), astreindre (19), bouillir (2), circonscrire (11), conclure (2), consentir (11), convaincre (2), corrompre (3), coudre (4), couvrir (10), desservir (2), dormir (3), dévêtir (3), enfuir (2), ensuivre (3), inclure (2), rire (2)
Macroclass 5	3rd conj.	accroître (4), apparaître (14), avoir (1), naître (3), savoir (1), être (1)
Macroclass 6	first conj.	aboyer (94), envoyer (2)
	3rd conj.	asseoir (1), boire (2), croire (2), entrevoir (3), prévoir (1), rasseoir (1), surseoir (1)

tions also found with some third conjugation verbs – but not in the infinitive.[11] The traditional second conjugation is so homogeneous that it is represented by only two microclasses, and their similarity with some verbs of the traditional third conjugation is large enough for them to cluster together. The verbs of the traditional third conjugation are split into different macroclasses, confirming that it has little internal homogeneity. Looking at the table, the clustering seems to be done on the basis of the infinitive ending. However, there was actually no primacy given to infinitive forms over any other in the evaluation

[11] The preference of our algorithm for this grouping is obviously due to the fact that there are many pairs of cells exhibiting a *X*wa∼*X*waj alternation, while fewer pairs of cells exhibit alternations typical of the first conjugation.

Macroclasses	Traditionnal	Lexemes
Macroclass 1	first conj.	abandonar (12), achar (3), chegar (20), desempenhar (4), ficar (911), ganhar (1), jogar (177), levar (162), nomear (53), pagar (1), passar (155), voar (17)
Macroclass 2	second conj.	adoecer (1), arder (1), combater (11), crer (2), decorrer (30), defender (42), doer (3), erguer (1), escrever (8), esquecer (2), perder (1), receber (90), resolver (7), valer (2)
Macroclass 4	3rd conj.	abrir (1), cair (11), cobrir (3), concluir (28), construir (6), desmentir (5), explodir (3), garantir (90), ouvir (1), partir (9), pedir (4), reabrir (1), reduzir (11), rir (2), seguir (45), subir (9)
Macroclass 4	second conj.	impor (17), ter (9)
	3rd conj.	intervir (1), vir (4)
Macroclass 5	first conj.	estar (1)
	second conj.	caber (1), condizer (2), fazer (5), haver (1), querer (2), ser (1), trazer (1), ver (5)

Table 13: Portuguese: Comparison of inferred macroclasses (local strategy) vs traditional conjugations

of inflectional behavior. In light of this classification, it seems that the local strategy does lead to a kind of inflectional classification close to that produced by descriptive morphologists, while diverging in terms of details from the extant standard classification by highlighting previously overlooked similarities between microclasses that are prevalent enough in paradigms to emerge as classificatory criteria.

The picture is similar for European Portuguese. The traditional account distinguishes between three conjugations based on the infinitive. The global strategy finds 13 macroclasses with little relation to the traditional classification. The local strategy finds five macroclasses, whose content is detailed in Table 13. The first three classes clearly match the three traditional conjugations, with characteristic theme vowels in *-a*, *-e*, and *-i*. The two remaining classes are not coherent in terms of theme vowels but have other notable properties. Macroclass 4 groups verbs with a stem alternant in *-ɲ* in the indicative past imperfective, in the subjunctive, and in the present indicative 1SG. This leads to a distinctive set of alternations that sets them apart from all other macroclasses, and has a stronger effect on classification than the theme vowel, which may be *-o*, *-e* or *-i*. Macroclass 5 groups

together a set of highly irregular verbs, and exhibits maximal dissimilarity for a cluster of such a small size (19 microclasses). There is no single reason for these microclasses to be grouped together, but there is definitely no strong reason as to why they should be placed somewhere else: all of them strongly depart from regular conjugations in one way or another.

All in all, then, we observe that, under the local strategy, our algorithm produces a classification that is strongly congruent with conventional practice, and highly defensible from a linguist's perspective, while being immune to some biases of grammatical tradition, such as that of giving stronger weight to citation forms than to other paradigm cells in deciding what should be grouped together.

5 CONCLUSION

This paper has presented a method for inferring inflection classes that captures crucial intuitions and heuristics used by descriptive linguists while being entirely systematic and unambiguously applicable to any system. Our modelling strategy is computational: we start from a few leading ideas on inflectional classification and propose a computational implementation of these ideas.

We started from a distinction between inflectional microclasses and macroclasses. A system of microclasses is based on identity of inflectional behavior across lexemes: two lexemes belong to the same microclass if and only if they exhibit exactly the same alternations. A system of macroclasses groups together microclasses exhibiting *similar* rather than *identical* behavior. Since similarity is gradual and multidimensional, there is no single agreed upon strategy to choose an appropriate system of macroclasses. Many authors rely on criteria such as productivity or regularity to that effect. We proposed to ground the choice of macroclasses solely in the direct examination of paradigms of surface forms. How such a form-based classification correlates with other forms of classification is an empirical question that is best addressed once the form-based generalizations are known.

With this goal in mind, we presented an algorithm that builds on the Minimum Description Length principle to explore partitions of the set of lexemes into classes. The underlying idea is that the optimal set of macroclasses for a system is the set that leads to the most compact

description of the system; this captures the intuition that macroclasses should help the linguist or language learner by minimizing the quantity of rote learning necessary to make sense of the system.

The algorithm was applied to two datasets of French and European Portuguese conjugation, under two different strategies for representing inflectional behavior: under the local strategy, inflectional behaviour is modelled by examining pairwise similarities and differences between paradigm cells of a lexeme, while under the global strategy, it is modelled by examining the similarities and differences that hold for the whole paradigm at once.

We find that the local segmentation better captures paradigmatic structure, and produces macroclass systems that resemble those elaborated by grammatical traditions. However, we also identify some previously unidentified macroclasses. We consider the differences between our classifications and those found in the literature to be attributable to a more principled view of classification. First, we confirm that unproductive and/or irregular microclasses do not cluster together in terms of formal similarity, and hence that grouping them together, as is usual in the French tradition, is unwarranted. Second, our model does not give any privileged status to the citation form, unlike what is usually done: hence the infinitive plays no privileged role in classification. Hence inflectional characteristics that are transparent from the infinitive form, such as theme vowels, play a role in the classification only inasmuch as they result in distinct alternation patterns. Third and finally, the implemented model is able to take into account all similarities and differences between all paradigm cells among dozens of macroclasses, a task whose manual execution is not feasable. This allows previously unobserved patterns of similarity to emerge.

We make no claim as to the importance of inflectional macroclasses as an analitycal tool. Our goal was rather to establish that it is possible to devise a systematic method of inference of macroclasses from raw paradigms. Of course, a partition of the lexicon into a small set of clusters of lexemes with similar behavior is one among a variety of ways one may approach the structure of an inflectional system; the fact that is has a longstanding tradition as a pedagogical tool is not reason enough not to explore alternative forms of classification. Beniamine and Bonami (2016) is an initial attempt at inferring from surface patterns lattice-shaped classifications such as those familiar

from Network Morphology (Brown and Hippisley 2012) and HPSG approaches to morphology (Bonami and Crysmann 2016).

APPENDIX — DESCRIPTIONS OF INFLECTION SYSTEMS

This appendix presents in some detail the class of inflection system models on which we rely for macroclass inference and description length assessment.

As mentioned in the Section 2.2.2, we are not interested in finding the shortest possible description, but rather in finding the way of clustering microclasses into macroclasses that produces the largest decrease in description length. Therefore, we only need to compute the contribution to the overall description length of those parts of the description that vary when the set of macroclasses varies. The description of the set of microclasses will be constant over all possible clustering of microclasses for a given system. We include it nevertheless in the description of the inflection class system so as to be able to compare different descriptions of the same system that use different strategies for alternation pattern inference, e.g. a global or a local strategy.

The description length we define below does not take into account the number of bits needed to declare each patterns and lexemes, the name of the cells and their pairing, the contexts in which patterns apply,[12] and the description of the procedure to decode the data. None of this will vary across competing partitions, so none of it is useful to us in selecting a partition.

Following Sagot and Walther (2011), we decompose the overall description length into a number of terms, each of which encoding a distinct part of the description. We define the description length of a given description D of an inflectional system as the sum of the description lengths of the four following components, which we briefly define below: microclasses, clusters, patterns and residue:

$$\mathrm{DL}(D) = \mathrm{DL}_M(D) + \mathrm{DL}_C(D) + \mathrm{DL}_P(D) + \mathrm{DL}_R(D).$$

[12] These contexts have been replaced by placeholders when abstracting patterns, but they could be stored and generalised as in (Bonami and Beniamine 2015), and the classes of applicability could be taken into account in the residual information (for which see below).

In the remainder of this appendix, we shall use the system presented in Section 3.1, Tables 8, 9 and 10 as a running example. Diagrams and explicit descriptions correspond to the description $D_{\{\{\text{AMENER, BOIRE}\},\{\text{DIRE}\}\}}$, which relies on the partition {{AMENER, BOIRE},{DIRE}} of the set of microclasses.

A.1 *Mapping microclasses to lexemes*

We define $\text{DL}_M(D)$ as the minimum number of bits needed to describe the mapping between lexemes and microclasses in description D. If we suppose that the set of lexemes $\mathscr{L}$ is ordered in a predefined way, such a mapping can be simply expressed as a list of $|\mathscr{L}|$ microclass identifiers that is parallel to the list of $|\mathscr{L}|$ lexemes.

Let us call $\mathscr{M}$ the set of microclass identifiers. If we define $\text{occ}(m)$ as the number occurrences of a given microclass identifier $m \in \mathscr{M}$, the description length $\text{DL}_M(D)$ of the "microclasses" section of the description D can be defined as follows:

$$\begin{aligned}\text{DL}_M(D) &= -|\mathscr{L}| \cdot \sum_{m \in \mathscr{M}} \frac{\text{occ}(m)}{|\mathscr{L}|} \cdot \log_2 \frac{\text{occ}(m)}{|\mathscr{L}|} \\ &= -\sum_{m \in \mathscr{M}} \text{occ}(m) \cdot \log_2 \frac{\text{occ}(m)}{|\mathscr{L}|}.\end{aligned}$$

Applying this definition to our running example, which contains three microclasses occurring once each, we obtain:

$$\text{DL}_M\left(D_{\{\{\text{AMENER, BOIRE}\},\{\text{DIRE}\}\}}\right) - 3 \log_2 \frac{1}{3} \approx 4.75$$

A.2 *Mapping microclasses to microclass clusters*

We can also assume that the set $\mathscr{M}$ of microclasses is associated with a predefined order. We can then express the mapping from microclasses to microclass clusters by simply listing microclass cluster identifiers following the same order (the i-th cluster identifier will indicate the cluster to which the i-th microclass belongs).

In a parallel way to the above, and defining the set of microclass clusters as $\mathscr{C}$, we can then write:

$$\text{DL}_C(D) = -\sum_{c \in \mathscr{C}} \text{occ}(c) \cdot \log_2 \frac{\text{occ}(c)}{|\mathscr{M}|}.$$

Note that the number of occurrences $\mathrm{occ}(c)$ of a cluster $c \in \mathscr{C}$ in the "clusters" part of the description corresponds to its size, i.e. the number of microclasses it contains.

Applying this definition to our running example, in which one cluster appears twice and the other appears only one time, we obtain:

$$\begin{aligned}\mathrm{DL}_C\left(D_{\{\{\textsc{amener, boire}\},\{\textsc{dire}\}\}}\right) &= -2\log_2\frac{2}{3} - \log_2\frac{1}{3}\\ &\approx 2.75\end{aligned}$$

Note that this result also holds for the other two partitions, the distribution of clusters is the same:

$$\begin{aligned}\mathrm{DL}_C\left(D_{\{\{\textsc{amener}\},\{\textsc{boire,dire}\}\}}\right) &= \mathrm{DL}_C\left(D_{\{\{\textsc{amener,dire}\},\{\textsc{boire}\}\}}\right)\\ &= \mathrm{DL}_C\left(D_{\{\{\textsc{amener, boire}\},\{\textsc{dire}\}\}}\right)\end{aligned}$$

DL_C is lower in descriptions with fewer, larger clusters, as less information is required for selecting the right cluster for each microclass. The extreme case is when there is only one cluster. In this case, the probability of this cluster is 1 and the corresponding value for DL_C is 0. Conversely, DL_C is higher when there are many smaller clusters:

$$\mathrm{DL}_C\left(D_{\{\{\textsc{amener, boire,dire}\}\}}\right) = -3\log_2\frac{3}{3} = 0$$

$$\mathrm{DL}_C\left(D_{\{\{\textsc{amener}\},\{\textsc{dire}\},\{\textsc{boire}\}\}}\right) = -3\log_2\frac{1}{3} \approx 4.75$$

A.3 *Relation between patterns and clusters*

For each pair of cells in the paradigm, the description associates clusters with alternation patterns used by lexemes in this cluster. This relation is not a function: several patterns can appear in a cluster, and several clusters can make use of a same pattern.

Let us call $\mathscr{K}$ the set of paradigm cells. $\mathscr{K}^2$ is then set of all n cell pairs, which we can assume is associated with a predefined order $\mathbf{k_1} \prec \mathbf{k_2} \prec \ldots \prec \mathbf{k_n}$. Let us refer to the set of alternation patterns identifiers as $\mathscr{P}$. The relation between patterns and clusters can then be encoded in the form of a sequence of pairs of the form (c, p), where $c \in \mathscr{C}$ is a cluster identifier and $p \in \mathscr{P}$ is an alternation pattern identifier. More precisely, since $\mathscr{C}$ is also supposed to be associated with a total

order, the relation between patterns and clusters can be provided as follows: first, all pairs (c, p) for the first cell pair $\mathbf{k_1}$ can be provided, ordered according to the cluster it includes; next, all pairs for $\mathbf{k_2}$ can be provided; the shift from $\mathbf{k_1}$ pairs to $\mathbf{k_2}$ pairs is visible because the cluster in the last $\mathbf{k_1}$ pair is the last cluster in (ordered) $\mathscr{C}$, whereas the cluster in the first $\mathbf{k_2}$ pair is the first cluster in $\mathscr{C}$; we then resume with $\mathbf{k_3}$ pairs, and so on.

Let us decompose $\mathrm{DL}_p(D)$ into the contribution $\mathrm{DL}_{p_c}(D)$ of cluster identifiers and the contribution $\mathrm{DL}_{p_p}(D)$ of pattern identifiers. Let us call $\mathrm{occ}_{\mathbf{k}}(c)$ (resp. $\mathrm{occ}_{\mathbf{k}}(p)$) the number of occurrences of a given cluster c (resp. of a given pattern p) in pairs of the form (c, p) associated with a given cell pair $\mathbf{k} \in \mathscr{K}$. Let us note N the total number of pairs of the form (c, p), i.e. $N = \sum_{c' \in \mathscr{C}} \mathrm{occ}_{\mathbf{k}}(c') = \sum_{p' \in \mathscr{P}} \mathrm{occ}_{\mathbf{k}}(p')$. The probability of occurrence of a given cluster identifier $c \in \mathscr{C}$ is then:

$$\begin{aligned} P(c) &= \sum_{\mathbf{k} \in \mathscr{K}^2} \frac{\mathrm{occ}_{\mathbf{k}}(c)}{\sum_{c' \in \mathscr{C}} \mathrm{occ}_{\mathbf{k}}(c')} \\ &= \frac{1}{N} \sum_{\mathbf{k} \in \mathscr{K}^2} \mathrm{occ}_{\mathbf{k}}(c). \end{aligned}$$

Therefore,

$$\begin{aligned} \mathrm{DL}_{p_c}(D) &= -N \sum_{c \in \mathscr{C}} P(c) \cdot \log_2 P(c) \\ &= -\sum_{c \in \mathscr{C}} \sum_{\mathbf{k} \in \mathscr{K}^2} \mathrm{occ}_{\mathbf{k}}(c) \cdot \log_2 \frac{\mathrm{occ}_{\mathbf{k}}(c)}{N} \end{aligned}$$

Similarly, he probability of occurrence of a given pattern identifier $p \in \mathscr{P}$ is:

$$\begin{aligned} P(p) &= \sum_{\mathbf{k} \in \mathscr{K}^2} \frac{\mathrm{occ}_{\mathbf{k}}(p)}{\sum_{p' \in \mathscr{P}} \mathrm{occ}_{\mathbf{k}}(p')} \\ &= \frac{1}{N} \sum_{\mathbf{k} \in \mathscr{K}^2} \mathrm{occ}_{\mathbf{k}}(p). \end{aligned}$$

Therefore,

$$\begin{aligned} \mathrm{DL}_{p_p}(D) &= -N \sum_{p \in \mathscr{P}} P(p) \cdot \log_2 P(p) \\ &= -\sum_{p \in \mathscr{P}} \sum_{\mathbf{k} \in \mathscr{K}^2} \mathrm{occ}_{\mathbf{k}}(p) \cdot \log_2 \frac{\mathrm{occ}_{\mathbf{k}}(p)}{N} \end{aligned}$$

The description length $DL_P(D) = DL_{P_c}(D) + DL_{P_p}(D)$ of the "patterns" section of the description can then be computed as:

$$DL_P(D) = -\sum_{\mathbf{k}\in\mathscr{K}^2}\left(\sum_{c\in\mathscr{C}}\mathrm{occ}_{\mathbf{k}}(c)\cdot\log_2\frac{\mathrm{occ}_{\mathbf{k}}(c)}{N} + \sum_{p\in\mathscr{P}}\mathrm{occ}_{\mathbf{k}}(p)\cdot\log_2\frac{\mathrm{occ}_{\mathbf{k}}(p)}{N}\right)$$

Applying this definition to our running example, we obtain:

$$\begin{aligned}\mathrm{DL}_P\left(D_{\{\{\text{AMENER, BOIRE}\},\{\text{DIRE}\}\}}\right) = &-2\log_2\frac{1}{2}\\ &-2\log_2\frac{1}{2}\\ &-3\log_2\frac{1}{3}\\ &-\log_2\frac{1}{3} - 2\log_2\frac{2}{3}\\ &-3\log_2\frac{1}{3}\\ &-\log_2\frac{1}{3} - 2\log_2\frac{2}{3}\\ \approx\ &14.26\end{aligned}$$

In the same fashion, we have:

$$\begin{aligned}\mathrm{DL}_P\left(D_{\{\{\text{AMENER}\},\ \{\text{BOIRE,DIRE}\}\}}\right) &= \mathrm{DL}_P\left(D_{\{\{\text{AMENER,DIRE}\},\{\text{BOIRE}\}\}}\right)\\ &= -10\log_2\frac{1}{3} - 8\log_2\frac{2}{3}\\ &\approx 20.52\end{aligned}$$

$$\begin{aligned}\mathrm{DL}_P\left(D_{\{\{\text{AMENER,BOIRE,DIRE}\}\}}\right) &= -2\log_2\frac{1}{2} - 6\log_2\frac{1}{3}\\ &\approx 11.5\end{aligned}$$

$$\begin{aligned}\mathrm{DL}_P\left(D_{\{\{\text{AMENER}\},\{\text{BOIRE}\},\{\text{DIRE}\}\}}\right) &= -16\log_2\frac{1}{3} - 2\log_2\frac{2}{3}\\ &\approx 26.52\end{aligned}$$

Unsurprisingly, the most efficient way to assign patterns to clusters is to have only one cluster, and the worst is to have as many clusters as microclasses.

A.4 *Residual ambiguity*

Since a cluster can be associated with several patterns for a same pair of cells, clustering can produce ambiguity. A complete description has to account for the information needed to disambiguate such ambiguities. As for the patterns, the necessary residual information is dispatched over each pair of cells. As it is internal to each cluster, it also has to be repeated for each of them.

Given a microclass cluster identifier $c \in \mathscr{C}$ and a pair of cells $\mathbf{k} \in \mathscr{K}^2$, the corresponding residual information is provided in the form of a set of pairs of the form (m, p), where $m \in \mathscr{M}$: such a pair means that the microclass m follows pattern p on cell pair $\mathbf{k}$. Of course, only those microclasses that belong to the cluster (identified by) c can and should be included. Since the list of microclasses included in c is a piece of information that has been already taken into account, and since microclasses have been ordered, the residual information of a given cluster c and a given cell pair $\mathbf{k} \in \mathscr{K}^2$ can be given in the form of a simple list of patterns, one for each microclass included in c, in the correct order. In such a list, each pattern p will occur with a probability $\mathrm{occ}^c_{\mathbf{k}}(p)/\mathrm{occ}(c)$, where $\mathrm{occ}^c_{\mathbf{k}}(p)$ is the number of microclasses in c that use pattern p for cell pair $\mathbf{k}$. We call $\mathscr{P}_{\mathbf{k}}(c)$ the set of patterns that are used my at least one microclass in cluster c for cell pair $\mathbf{k}$.

As a result:

$$DL(R) = \sum_{c \in \mathscr{C}} \sum_{\mathbf{k} \in \mathscr{K}^2} \sum_{p \in \mathscr{P}_{\mathbf{k}}(c)} \mathrm{occ}^c_{\mathbf{k}}(p) \cdot \log \frac{\mathrm{occ}^c_{\mathbf{k}}(p)}{\mathrm{occ}(c)}.$$

In the example above, in the first cluster, for each of the two ambiguous cells, each of the two patterns happens for only one microclass.

$$\mathrm{DL}_R(D_{\{\{\text{AMENER, BOIRE}\},\{\text{DIRE}\}\}}) = -4 \log_2 \frac{1}{2} = 4.$$

We also have:

$$\begin{aligned}\mathrm{DL}_R(D_{\{\{\text{AMENER}\},\ \{\text{BOIRE,DIRE}\}\}}) &= \mathrm{DL}_R(D_{\{\{\text{AMENER,DIRE}\},\{\text{BOIRE}\}\}}) \\ &= -6 \log_2 \frac{1}{2} = 6\end{aligned}$$

$$\mathrm{DL}_R(D_{\{\{\text{AMENER,BOIRE,DIRE}\}\}}) = -2 \log_2 \frac{2}{3} - 7 \log_2 \frac{1}{3} \approx 12.26$$

$$\mathrm{DL}_R(D_{\{\{\text{AMENER}\},\{\text{BOIRE}\},\{\text{DIRE}\}\}}) = 0$$

Unsurprisingly, while clustering maximally tends to decrease DL_P, it tends to increase ambiguity and thus DL_R, while having smaller clusters leads to less ambiguity, thus a smaller $DL(R)$. In minimizing the total description length, we seek an balance between these measures.

We can now gather all the partial DLs in Table 14 to compare each classification and recognise {{AMENER, BOIRE},{DIRE}} as the best partition according to description length.

Table 14: Description lengths for all the possible classifications of Table 8 in microclasses and macroclasses

Partition	$DL(M)$	$DL(C)$	$DL(\mathscr{P})$	$DL(R)$	total DL
{{AMENER}, {BOIRE,DIRE}}	4.75	2.75	20.52	6	34.01
{{AMENER, BOIRE},{DIRE}}	4.75	2.75	14.26	4	25.75
{{AMENER, DIRE},{BOIRE}}	4.75	2.75	20.52	6	34.01
{{AMENER, BOIRE,DIRE}}	4.75	0	11.50	12.26	28.5
{{AMENER}, {DIRE},{BOIRE}}	4.75	4.75	26.52	0	36.01

REFERENCES

Farrell ACKERMAN, James P. BLEVINS, and Robert MALOUF (2009), Parts and wholes: implicative patterns in inflectional paradigms, in James P. BLEVINS and Juliette BLEVINS, editors, *Analogy in Grammar*, pp. 54–82, Oxford University Press, Oxford.

Farrell ACKERMAN and Robert MALOUF (2013), Morphological organization: The low conditional entropy conjecture., *Language*, 89(3):429–464, doi:10.1353/lan.2013.0054.

Malin AHLBERG, Markus FORSBERG, and Manstio HULDEN (2014), Semi-supervised learning of morphological paradigms and lexicons, in *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, Gothenburg, Sweden 26–30 April 2014*, pp. 569–578, ISBN 978-1-937284-78-7, doi:10.3115/v1/E14-1060.

Adam ALBRIGHT and Bruce HAYES (2003), Rules vs. analogy in English past tenses: A computational/experimental study, *Cognition*, 90:119–161, doi:10.1016/S0010-0277(03)00146-X.

Adam ALBRIGHT and Bruce HAYES (2006), Modeling productivity with the gradual learning algorithm: The problem of accidentally exceptionless generalizations, *Gradience in grammar: Generative perspectives*, pp. 185–204.

Mark ARONOFF (1994), *Morphology by Itself: Stems and Inflectional Classes*, Linguistic inquiry monographs, MIT Press, ISBN 9780262510721.

Sacha BENIAMINE (2017), Une approche universelle pour l'abstraction automatique d'alternances morphophonologiques, in *Traitement Automatique des Langues Naturelles (TALN)*, Association pour le Traitement Automatique des Langues (ATALA), pp. 77–85.

Sacha BENIAMINE and Olivier BONAMI (2016), A comprehensive view on inflectional classification, paper presented at the *Annual Meeting of the Linguistic Association of Great Britain*, Paris.

James P. BLEVINS (2005), Word-based declensions in Estonian, in Geert E. BOOIJ and Jaap van MARLE, editors, *Yearbook of Morphology 2005*, pp. 1–25, Springer.

James P. BLEVINS (2006), Word-based morphology, *Journal of Linguistics*, 42:531–573, ISSN 1469-7742, doi:10.1017/S0022226706004191.

Olivier BONAMI (2014), La structure fine des paradigmes de flexion, Habilitation à diriger des recherches, Université Paris Diderot.

Olivier BONAMI and Sacha BENIAMINE (2015), Implicative structure and joint predictiveness, in Vito PIRELLI, Claudia MARZI, and Marcello FERRO, editors, *Word Structure and Word Usage. Proceedings of the NetWordS Final Conference*.

Olivier BONAMI and Sacha BENIAMINE (2016), Joint predictiveness in inflectional paradigms, *Word Structure*, 9(2):156–182.

Olivier BONAMI and Gilles BOYÉ (2014), De formes en thèmes, in Florence VILLOING, Sarah LEROY, and Sophie DAVID, editors, *Foisonnements morphologiques. Etudes en hommage à Françoise Kerleroux*, pp. 17–45, Presses Universitaires de Paris Ouest.

Olivier BONAMI, Gilles BOYÉ, Hélène GIRAUDO, and Madeleine VOGA (2008), Quels verbes sont réguliers en français?, in *Actes du premier Congrès Mondial de Linguistique Française*, pp. 1511–1523, doi:10.1051/cmlf08186.

Olivier BONAMI, Gauthier CARON, and Clément PLANCQ (2014), Construction d'un lexique flexionnel phonétisé libre du français, in Franck NEVEU, Peter BLUMENTHAL, Linda HRIBA, Annette GERSTENBERG, Judith MEINSCHAEFER, and Sophie PRÉVOST, editors, *Actes du quatrième Congrès Mondial de Linguistique Française*, pp. 2583–2596, doi:10.1051/shsconf/20140801223.

Olivier BONAMI and Berthold CRYSMANN (2016), The role of morphology in constraint-based lexicalist grammars, in Andrew HIPPISLEY and Gregory T. STUMP, editors, *Cambridge Handbook of Morphology*, pp. 609–656, Cambridge University Press, Cambridge.

Olivier BONAMI and Ana R. LUÍS (2014), Sur la morphologie implicative dans la conjugaison du portugais : une étude quantitative, in Jean-Léonard LÉONARD, editor, *Morphologie flexionnelle et dialectologie romane. Typologie(s) et modélisation(s)*, number 22 in Mémoires de la Société de Linguistique de Paris, pp. 111–151, Peeters, Leuven.

Dunstan BROWN (1998), *From the general to the exceptional*, Ph.D. thesis, University of Surrey.

Dunstan BROWN and Roger EVANS (2012), Morphological complexity and unsupervised learning: validating Russian inflectional classes using high frequency data, in Kiefer FERENCE, Mária LADÁNYI, and Péter SIPTÁR, editors, *(Ir)regularity, analogy and frequency, selected papers from the 14th International morphology meeting, Budapest, 13–16 May 2010*, Current Issues in Morphological Theory, pp. 135–162, John Benjamins Publishing Co., Amsterdam, doi:10.1075/cilt.322.07bro.

Dunstan BROWN and Andrew HIPPISLEY (2012), *Network Morphology: A Defaults-based Theory of Word Structure*, Cambridge Studies in Linguistics, Cambridge University Press, ISBN 9781107005747, doi:10.1017/CBO9780511794346.

Andrew D. CARSTAIRS (1987), *Allomorphy in Inflexion*, Croom Helm linguistics series, Croom Helm, ISBN 9780709934837.

Andrew CARSTAIRS-MCCARTHY (1994), Inflection Classes, Gender, and the Principle of Contrast, *Language*, 70(4):737–788, ISSN 00978507, doi:10.2307/416326.

Rudi L. CILIBRASI and Paul M. B. VITANYI (2005), Clustering by Compression, *IEEE Transactions on Information Theory*, 51(4):1523–1545, doi:10.1109/tit.2005.844059, http://dx.doi.org/10.1109/TIT.2005.844059.

Harald CLAHSEN (2006), Dual-mechanism morphology, in Keith BROWN, editor, *Encyclopedia of Language and Linguistics*, volume 4, pp. 1–5, Elsevier.

Greville G. CORBETT (1982), Gender in Russian: an account of gender specification and its relationship to declension, *Russian Linguistics*, 2:197–232.

Greville G. CORBETT (2009), Canonical Inflectional Classes, in Fabio MONTERMINI, Gilles BOYÉ, and Jesse TSENG, editors, *Selected Proceedings of the 6th Décembrettes: Morphology in Bordeaux*, volume 1-11, Cascadilla Proceedings Project, Somerville, MA, USA.

Greville G. CORBETT and Norman M. FRASER (1993), Network Morphology: a DATR account of Russian nominal inflection, *Journal of Linguistics*, 29:113–142, ISSN 1469-7742, doi:10.1017/S0022226700000074.

Wolfgang U DRESSLER, Marianne KILANI-SCHOCH, Natalia GAGARINA, Lina PESTAL, and Markus PÖCHTRAGER (2008), On the Typology of Inflection Class Systems, *Folia Linguistica*, 40(1-2):51–74, doi:10.1515/flin.40.1-2.51.

Wolfgang U. DRESSLER, Willi MAYERTHALER, Oswald PANAGL, and Wolfgang Ullrich WURZEL (1987), *Leitmotifs in natural morphology*, volume 10, John Benjamins Publishing, doi:10.1075/slcs.10.

Wolfgang U. DRESSLER and Anna M. THORNTON (1996), Italian Nominal Inflection, *Wiener Linguistische Gazette*, 55-57:1–26.

Markus DREYER and Jason EISNER (2011), Discovering Morphological Paradigms from Plain Text Using a Dirichlet Process Mixture Model, in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pp. 616–627, Association for Computational Linguistics, Stroudsburg, PA, USA, ISBN 978-1-937284-11-4.

Greg DURRETT and John DENERO (2013), Supervised Learning of Complete Morphological Paradigms, in *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1185–1195, Association for Computational Linguistics, Atlanta, Georgia.

Ramy ESKANDER, Nizar HABASH, and Owen RAMBOW (2013), *Automatic Extraction of Morphological Lexicons from Morphologically Annotated Corpora*, Association for Computational Linguistics, Seattle, Washington, USA.

John GOLDSMITH (2001), Unsupervised Learning of the Morphology of a Natural Language, *Computational Linguistics*, 27(2):153–198, ISSN 0891-2017, doi:10.1162/089120101750300490.

John GOLDSMITH and Jeremy O'BRIEN (2006), Learning inflectional classes, *Language Learning and Development*, 24(4):219–250, doi:10.1207/s15473341lld0204_1.

Peter D. GRÜNWALD (2007), *Minimum Description Length Principle*, MIT press, Cambridge, MA, ISBN 978-0-262-07281-6.

Marianne KILANI-SCHOCH and Wolfgang U. DRESSLER (2005), *Morphologie naturelle et flexion du verbe français*, Tübinger Beiträge zur Linguistik, G. Narr, ISBN 9783823361619.

Jackson LEE and John A. GOLDSMITH (2013), Automatic morphological alignment and clustering, presented at the 2nd American International Morphology Meeting.

Robert MALOUF (in press), Abstractive morphological learning with a recurrent neural network, *Morphology*, 27(4):431–458.

Peter H. MATTHEWS (1972), *Inflectional morphology: A theoretical study based on aspects of Latin verb conjugation*, Cambridge University Press.

Petar MILIN, Dušica FILIPOVIĆ ĐURĐEVIĆ, and Fermin MOSCOSO DEL PRADO MARTÍN (2009), The simultaneous effects of inflectional paradigms and classes on lexical recognition: Evidence from Serbian, *Journal of Memory and Language*, 60:50–64.

Christian MONSON, Alon LAVIE, Jaime CARBONELL, and Lori LEVIN (2004), Unsupervised Induction of Natural Language Morphology Inflection Classes, in

Proceedings of the Seventh Meeting of the ACL Special Interest Group in Computational Phonology (SIGPHON '04), pp. 52–61, doi:10.3115/1622153.1622160.

Fabio Montermini and Gilles Boyé (2012), Stem relations and inflection class assignment in Italian, *Word Structure*, 5:69–87.

Boris New, Christophe Pallier, Ludovic Ferrand, and Rafael Matos (2001), Une base de données lexicales du français contemporain sur internet: LEXIQUE., *L'année psychologique*, 101(3):447–462.

Garrett Nicolai, Colin Cherry, and Grzegorz Kondrak (2015), Inflection Generation as Discriminative String Transduction, in *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 922–931, Association for Computational Linguistics, Denver, Colorado, doi:10.3115/v1/N15-1093.

Marc Plénat (1987), Morphologie du passé simple et du passé composé des verbes de l' "autre" conjugaison, *ITL Review of Applied Linguistics*.

Jorma Rissanen (1978), Modeling by shortest data description, *Automatica*, 14:465–658.

Jorma Rissanen (1984), Universal coding, information, prediction, and estimation, *IEEE Tr. on Info. Th.*, 30(4):629–636, doi:10.1109/TIT.1984.1056936.

Benoît Sagot and Géraldine Walther (2011), Non-canonical inflection : data, formalisation and complexity measures., in Cerstin Mahlow and Michael Piotrowski, editors, *Systems and Frameworks in Computational Morphology*, volume 100, pp. 23–45, Springer-Verlag, Zurich, Switzerland.

Benoît Sagot and Géraldine Walther (2013), Implementing a formal model of inflectional morphology, in Cerstin Mahlow and Michael Piotrowski, editors, *Actes du Third International Workshop on Systems and Frameworks for Computational Morphology (SFCM 2013)*, volume 380 of *Communications in Computer and Information Science (CCIS)*, pp. 115–134, Humboldt-Universität, Springer-Verlag, Berlin, Germany.

Claude E. Shannon (1948), A Mathematical Theory of Communication, *Bell System Technical Journal*, 27(3):379–423, doi:10.1002/j.1538-7305.1948.tb01338.x, `http://dx.doi.org/10.1002/j.1538-7305.1948.tb01338.x`.

Robert R. Sokal and Charles D. Michener (1958), A statistical method for evaluating systematic relationships, *University of Kansas Scientific Bulletin*, 28:1409–1438.

Andrew Spencer (2012), Identifying stems, *Word Structure*, 5:88–108.

Gregory Stump and Raphael A. Finkel (2013), *Morphological Typology: From Word to Paradigm*, Cambridge Studies in Linguistics, Cambridge University Press, ISBN 9781107029248, doi:10.1017/CBO9781139248860.

Arlindo VEIGA, Sara CANDEIAS, and Fernando PERDIGÃO (2013), Generating a pronunciation dictionary for European Portuguese using a joint-sequence model with embedded stress assignment, *Journal of the Brazilian Computer Society*, 19(2):127–134, ISSN 0104-6500, doi:10.1007/s13173-012-0088-0.

João VERÍSSIMO and Harald CLAHSEN (2014), Variables and similarity in linguistic generalization: Evidence from inflectional classes in Portuguese, *Journal of Memory and Language*, 76:61–79.

Géraldine WALTHER (2013), *On canonicity in morphology: an empirical, formal and computational approach*, Ph.D. thesis, Université Paris Diderot.

Géraldine WALTHER (2016), Paradigm Realisation and the Lexicon, in Ferenc KIEFER, James P. BLEVINS, and Huba BARTOS, editors, *Morphological paradigms and functions*, Brill, Leiden, Pays-Bas.

Géraldine WALTHER and Benoît SAGOT (2011), Modeling and implementing non canonical morphological phenomena, *TAL*, 52(2):91–122.

Wolfgang Ulrich WURZEL (1984), *Flexionsmorphologie und Natürlichkeit. Ein Beitrag zur morphologischen Theoriebildung*, Akademie-Verlag, Berlin, translated as Wurzel (1989).

Wolfgang Ulrich WURZEL (1989), *Inflectional Morphology and Naturalness*, Kluwer, Dordrecht.

A syntax-semantics interface for Tree-Adjoining Grammars through Abstract Categorial Grammars

Sylvain Pogodalla
INRIA, Villers-lès-Nancy, France
Université de Lorraine, LORIA, Vandœuvre-lès-Nancy, France
CNRS, LORIA, Vandœuvre-lès-Nancy, France

ABSTRACT

We present a model of the syntax-semantics interface for Tree-Adjoining Grammars (TAGs). It is based on the encoding of TAGs within the framework of Abstract Categorial Grammars (ACGs). This encoding relies on a functional interpretation of the substitution and adjunction operations of TAGs. In ACGs, the abstract terms representing derivation trees are full-fledged objects of the grammar. These terms are mapped onto logical formulas representing the semantic interpretation of natural language expressions that TAGs can analyze. Because of the reversibility properties of ACGs, this provides a way to parse and generate with the same TAG-encoded grammar. We propose several analyses, including for long-distance dependencies, quantification, control and raising verbs, and subordinate clauses. We also show how this encoding easily extends to other phenomena such as idioms or scope ambiguities. All the lexical data for theses modellings are provided and can be run with the ACG toolkit, a software package dedicated to the development of ACGs that can use these grammars both for parsing and generation.

Keywords: Tree-Adjoining Grammars, syntax-semantics interface, Abstract Categorial Grammars

1 MOTIVATIONS

1.1 *Tree-Adjoining Grammar and semantic representation*

The Tree-Adjoining Grammar (TAG) formalism (Joshi *et al.* 1975; Joshi and Schabes 1997) is a formalism dedicated to the modelling of

natural languages. As the name indicates, the primary objects it considers are trees rather than strings, contrary to, for instance, context-free grammars. As such, the object language a TAG generates is a tree language, the language of the *derived trees*. These trees result from the application of two operations, *substitution* and *adjunction*, to a set of generators: the *elementary* trees. The substitution operation consists in replacing one leaf of a tree by another tree, while the adjunction operation consists in inserting a tree into another one by replacing an internal node with a whole tree. A sequence of such operations and the elementary trees they operate on can be recorded as a *derivation tree*. Reading the leaves of the derived tree, or computing the *yield*, produces the associated generated string language.

The class of the generated string languages strictly includes the one generated by context-free grammars. This property, together with other ones such as the crucial polynomial parsing property, plays an important role in the characterization of the expressive power that natural language modelling requires. Joshi (1985) proposed to call the class of languages (resp. grammars) necessary for describing natural languages the class of *mildly context-sensitive languages* or mCSL (resp. *mildly context-sensitive grammars* or mCSG). These formal and computational properties have been extensively studied[1] and provide TAG with appealing features for natural language processing. In addition to its formal properties, TAG has also been studied both from the perspective of fine-grained modellings of a wide range of linguistic phenomena, and from the perspective of large coverage. Large scale grammars have been developed for several languages, including English (XTAG Research Group 2001) and French (Abeillé 2002; Crabbé 2005; de La Clergerie 2005). In addition to these hand-crafted grammars, automatic extraction of TAGs has also been proposed (Xia *et al.* 2000; Xia 2001; Chen *et al.* 2006).

Another key feature that makes TAG relevant to natural language modelling lies in the capability of its elementary trees to locally specify (syntactic and semantic) dependencies between parts that can occur arbitrarily far from each other at the surface level at the end of a

[1] See for instance Vijay-Shanker and Joshi (1985), Vijay-Shanker (1987), Weir (1988), Kuhlmann and Möhl (2007), Kanazawa (2008b), Kanazawa (2008a), and Kallmeyer (2010).

derivation. This property to locally state, within the elementary trees, dependency constraints is also known as the *extended domain of locality* (Joshi 1994). Thanks to the adjunction operation, a dependency described locally in an elementary tree can end as a long-distance dependency in the resulting derived tree. The relevant structure to store the relations between the elementary trees that are used in a derivation is then the derivation tree. This makes the latter structure appropriate to derive semantic representations for TAGs.

It was however noticed that derivation trees do not directly express the semantic dependencies, and that they seem to lack some structural information (Vijay-Shanker 1992; Candito and Kahane 1998). To overcome this problem, several approaches have been proposed. Some rely on extensions of the TAG formalism (Rambow *et al.* 1995, 2001); some others revisit the derivation tree definition in order to allow for recovering all the semantic dependency relations (Schabes and Shieber 1994; Shieber 1994; Kallmeyer 2002; Joshi *et al.* 2003; Kallmeyer and Joshi 2003). However, solutions to the problem strictly relying on derivation trees have also been proposed. They make use of unification (Kallmeyer and Romero 2004, 2008), functional tree interpretation (Pogodalla 2004a, 2009), synchronous grammars (Nesson and Shieber 2006; Nesson 2009), or tree transduction (Shieber 2006; Kallmeyer and Kuhlmann 2012; Shieber 2014).

1.2 *TAG and Abstract Categorial Grammars: our approach*

In this article, we elaborate on Pogodalla (2004a, 2009) in order to propose a syntax-semantics interface and a semantic construction process for TAGs. We base our analysis on the framework of Abstract Categorial Grammars (ACGs: de Groote 2001). ACGs derive from type-theoretic grammars in the tradition of Lambek (1958), Curry (1961), and Montague (1973). They can be considered as a framework in which several grammatical formalisms may be encoded (de Groote and Pogodalla 2004), in particular TAGs (de Groote 2002). The definition of an ACG is based on a small set of mathematical primitives from type-theory, λ-calculus, and linear logic. These primitives combine via simple composition rules, offering ACGs a good flexibility. In particular, ACGs generate languages of linear λ-terms, which generalize both string and tree languages.

But ACGs are not restricted to languages of λ-terms encoding strings or trees. They can express logic-based semantic representation languages. And moving from one kind to another kind of language is realized by composing ACGs. We take advantage of the different composition modes to control the admissible derivation structures on the one hand, and to model the syntax-semantics interface on the other hand.

The core contribution of this article is to show that ACGs offer a suitable model of the syntax-semantics interface for TAG. By construction, this model is fully compositional and satisfies the homomorphic requirement between parse structures (terms representing derivation trees) and semantic terms. It relies on an encoding of TAGs into ACGs. For a given TAG G, with this encoding, we can construct and relate several ACGs that generate the same string language, derived tree language, and derivation tree language as G. By ACG composition, this encoding is the same as the one proposed by de Groote (2002) (that only addresses the syntactic encoding of TAG into ACG, not the syntax-semantics interface), which ensures the correctness of the (syntactic) encoding. This encoding corresponds to the path with solid lines from *TAG derivation trees* to *Strings* in Figure 1. But we introduce an intermediate level, of generalized derivations, on which we base our syntax-semantics interface (the dashed lines in Figure 1). Doing so, we separate the level required for transferring the syntactic structures into semantics, and vice-versa, from the level that controls those structures so that only the ones that TAG considers to be admissible (i.e., TAG derivations) are kept. We show that this level allows us to account for the semantics of long-distance dependencies, quantification, separate modification without multiple adjunction, control verbs, raising verbs, etc. Moreover, this is done in a principled way, following the standard homomorphism between the syntactic and semantic categories of Montague (1973).

Contrary to Pogodalla (2004a) and Kallmeyer and Romero (2004, 2008), and similarly to synchronous TAG analyses (Nesson and Shieber 2006; Nesson 2009), the semantic modelling we propose does not rely on an intermediate underspecification language. We show instead that this is not required in order to model long-distance dependencies, raising verbs, or quantification. We also introduce and precisely describe the syntax-semantics modelling for adjectives (with-

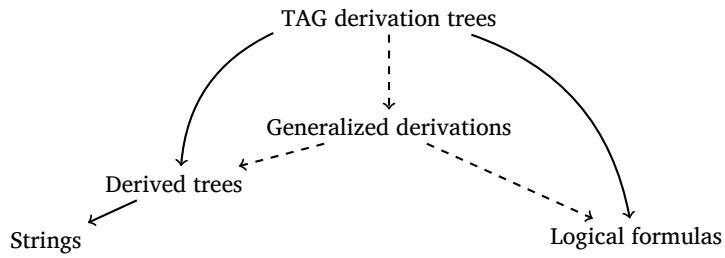


Figure 1: Overall architecture for the syntax-semantics interface

out multiple adjunctions), control verbs, and subordinate clauses. We also discuss the encoding of features. While these modellings can essentially be rephrased in synchronous TAG (and vice-versa:[2] the solid lines of Figure 1 also correspond to the synchronous TAG architecture for the syntax-semantics interface), it is not the case for some other ones, and we show how the approach easily extends, without requiring the design of new parsing algorithms, to other phenomena such as idioms[3] or subordinate clauses, for which we propose a novel modelling. Other TAG extensions such as the cosubstitution operation proposed by Barker (2010) to model scope ambiguities also easily fall within the scope of our approach and can be given a type-raising account. In particular, this account exemplifies how to model the non-functional nature of the form-meaning relation.

Finally, except for the type-raising account of quantification, the ACG model for the syntax-semantics interface of TAG that we propose belongs to the class of second-order ACGs. This class has the property that whatever the language we parse (strings, trees, or any kind of terms, such as first-order or higher-order logical formulas), parsing is polynomial. This parsing is implemented in the ACG toolkit.[4] Consequently, there is a parser that can actually recover the TAG derivation structure (if any) of some string, or of some derived tree, and interpret it as a logical formula, or that can actually recover the TAG derivation structure (if any) of some logical formula and interpret it as a derived

[2] Synchronous TAG analyses often hinge on Multi-Component TAG (MCTAG: Weir 1988), which is beyond the scope of this article. But we do not consider this to be an essential difference, since this can be integrated into the ACG approach as well (Pogodalla 2009). We discuss the differences between the approaches in Section 9.

[3] This encoding is due to Kobele (2012).

[4] The toolkit is available at `http://acg.loria.fr`.

tree or a string. The ACG framework is *inherently reversible* (Dymetman 1994), and parsing and generation of second-order ACGs are performed in polynomial time, including for the modellings that go beyond TAG (except the type-raising account of quantification), without having to design new parsers. Note, however, that we do not yet address the problem of logical-form equivalence (Shieber 1993) which states that, even if two formulas are logically equivalent, it might be possible to recover a derivation structure for one but not for the other.

We also validated the modellings and the lexicons we provide in this article, both for parsing and generation, by implementing in the ACG toolkit all the examples of this article. This results in a toy grammar (corresponding to about forty elementary trees) exemplifying the analyses of various linguistic phenomena presented in the article.[5] An extension to a real-size TAG grammar for French is ongoing.

1.3 *Organisation of the article*

In Section 2, we show a functional interpretation of the substitution and adjunction operations. We review the definitions that are necessary to model strings and trees as λ-terms (Section 2.2) and we use them to model the elementary trees of TAG. Section 3 reviews the definitions that are specific to ACGs. We show their main composition models and present their formal properties.

In Section 4, we introduce the ACGs that are necessary for the encoding of the syntax-semantics interface: the ACG relating strings and derived trees in Section 4.1; the ACG relating derived trees and generalized derivation trees in Section 4.2 and Section 4.3. We show that these generalized derivations over-generate with respect to TAG derivation trees: the full TAG encoding is not yet completed, but we already have all the necessary parts to implement the syntax-semantics interface.

Section 5 is devoted to the model of the syntax-semantics interface we propose. We first define the semantic representation language in Section 5.1. Then, in Section 5.2, we show how to interpret the

[5] The example files are available at `https://hal.inria.fr/hal-01242154/file/acg-examples.zip`. The script file illustrates the terms we use in this article and refers in comments to the relevant sections, equations, and term names.

generalized derivations as semantic terms and we provide several classical examples.

In Section 6 we complete the faithful TAG encoding by controlling the generalized derivations so that only TAG derivations are accepted. The correctness of the encoding is ensured by recovering, by ACG composition, de Groote's (2002) encoding. Then, again by ACG composition, we directly obtain a syntax-semantics interface for TAG. Because ACGs do not make use of features, we explain how we model the adjunction constraints induced by feature structures in TAG (Section 7).

In Section 8, we take advantage of the architecture we propose and give examples of modellings that this framework offers and that are beyond TAG. We finally discuss the relation of our approach to the syntax-semantics interface for TAG with other ones, in particular the ones using synchronous grammars or feature unification (Section 9).

2 BACKGROUND

2.1 *Adjunction and substitution*

A TAG consists of a finite set of elementary trees whose nodes are labelled by terminal and non-terminal symbols. Nodes labelled with terminals can only be leaves. Elementary trees are divided into *initial* and *auxiliary* trees. Figure 2 exemplifies such trees. Substituting α_{John} in α_{sleeps} consists in replacing a leaf of α_{sleeps} labelled with a non-terminal symbol NP with the tree α_{John} whose root node is labelled by NP as well.[6] Figure 3(a) shows an example of such a substitution (at Gorn address 1) and of its result. The corresponding derivation tree recording the substitution is represented in Figure 3(b), where

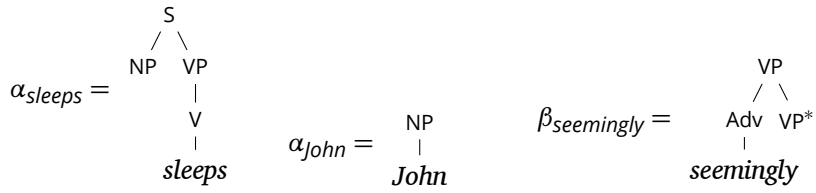


Figure 2: TAG elementary trees

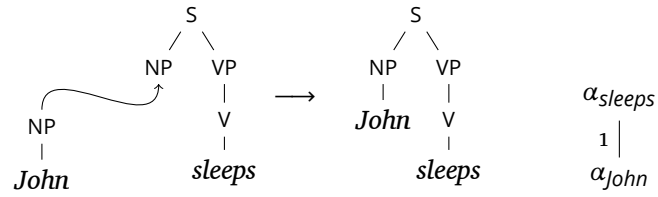
(a) Initial tree with one substitution node

(b) Initial tree with no substitution node

(c) Auxiliary tree

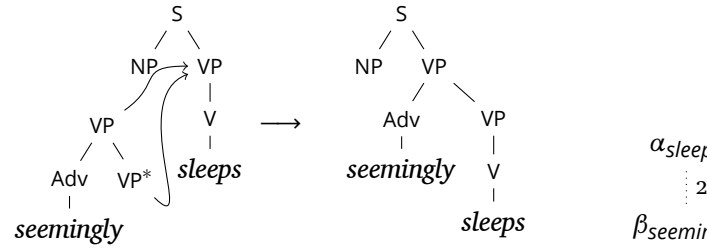
[6] Substitution sites are often marked by decorating the label with a ↓ symbol.

Figure 3: Substitution operation



(a) Substitution (b) Derivation tree

Figure 4: Adjunction operation



(a) Adjunction (b) Derivation tree

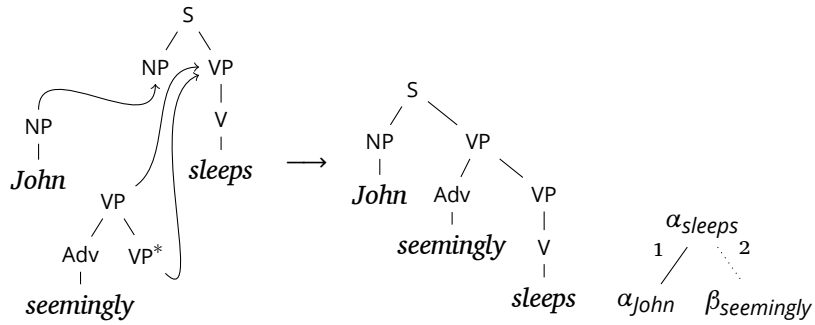
the Gorn address labels the edge between the two nodes, each of them being labelled by the name of the trees. Only initial trees, that possibly underwent some substitution or adjunction operations, can substitute into a leaf.

The adjunction of $\beta_{seemingly}$ into α_{sleeps} consists in inserting the tree $\beta_{seemingly}$ at the VP node of α_{sleeps}: the subtree of α_{sleeps} rooted at its VP node is first removed then substituted to the VP foot node of $\beta_{seemingly}$ (the leaf with the same label as the root and marked with *). The whole resulting tree is then plugged again at the VP node of α_{sleeps}, as Figure 4(a) shows. The associated derivation tree of Figure 4(b) records the adjunction with a dotted edge. Only auxiliary trees, that possibly underwent some substitution or adjunction operations, can adjoin into another tree.

Figure 5 shows a TAG analysis of *John seemingly sleeps*, which involves both operations, and the associated derivation tree.

2.2 *TAG elementary trees as functions*

We now present the two operations of adjunction and substitution using a functional interpretation of the elementary trees. We use the



(a) Adjunction and substitution

(b) Derivation tree

Figure 5: TAG analysis of *John seemingly sleeps*

standard notations of the typed λ-calculus and we formally present the syntax of λ-terms and their types.

Definition 1 (Types)**.** Let A be a set of atomic types. The set $\mathscr{T}(A)$ of *implicative types* built upon A is defined with the following grammar:

$$\mathscr{T}(A) ::= A \mid \mathscr{T}(A) \multimap \mathscr{T}(A) \mid \mathscr{T}(A) \to \mathscr{T}(A)$$

The set of *linear implicative types* built upon A is defined with the following grammar:

$$\mathscr{T}^{o}(A) ::= A \mid \mathscr{T}^{o}(A) \multimap \mathscr{T}^{o}(A)$$

Definition 2 (Higher-order signatures)**.** A *higher-order signature* Σ is a triple $\Sigma = \langle A, C, \tau \rangle$ where:

- A is a finite set of atomic types;
- C is a finite set of constants;
- $\tau : C \to \mathscr{T}(A)$ is a function assigning types to constants.

A higher-order signature $\Sigma = \langle A, C, \tau \rangle$ is *linear* if the codomain of τ is $\mathscr{T}^{o}(A)$.

Definition 3 (λ-Terms)**.** Let X be a countably infinite set of λ-variables. The set $\Lambda(\Sigma)$ of λ-terms built upon a higher-order signature $\Sigma = \langle A, C, \tau \rangle$ is inductively defined as follows:

- if $c \in C$ then $c \in \Lambda(\Sigma)$;
- if $x \in X$ then $x \in \Lambda(\Sigma)$;
- if $x \in X$ and $t \in \Lambda(\Sigma)$ and x occurs free in t exactly once, then $\lambda^{o} x.t \in \Lambda(\Sigma)$;

- if $x \in X$ and $t \in \Lambda(\Sigma)$, then $\lambda x.t \in \Lambda(\Sigma)$;
- if $t, u \in \Lambda(\Sigma)$ then $(t\,u) \in \Lambda(\Sigma)$.

Note there is a linear λ-abstraction (denoted by λ^o) and a (usual) intuitionistic λ-abstraction (denoted by λ). A variable that is bound by λ^o occurs exactly once in the body of the abstraction, whereas it can occur zero, one, or any number of times when it is bound by λ. This distinction is important when discussing the complexity of parsing with ACGs.

We also use the usual notions of α, β, and η conversions (Barendregt 1984), as well as the left associativity of (linear and non-linear) application (so $(t\ u)\ v$ is written $t\ u\ v$), the right associativity of (linear and non-linear) abstraction over several variables (so $\lambda x.(\lambda y.(\lambda z.t)) = \lambda x\ y\ z.t = \lambda x\ y.\lambda z.t$, etc.; the same for λ^o), and the right associativity of implication (so $\alpha \to (\beta \to \gamma) = \alpha \to \beta \to \gamma$; the same for $\multimap$).

Definition 4 (Typing judgment)**.** Given a higher-order signature Σ, the typing rules are given with an inference system whose judgments are of the form: $\Gamma; \Delta \vdash_\Sigma t : \alpha$ where:

- Γ is a finite *set* of non-linear variable typing declarations of the form $x : \beta$ where x is a variable and β is a type;
- Δ is a finite *multi-set* of linear variable typing declarations of the form $x : \beta$ where x is a variable and β is a type. In order to distinguish the elements of the typing declaration, we always use variables with different names.

Both Γ and Δ may be empty. If both of them are empty, we usually write $t : \alpha$ (t is of type α) instead of $\vdash_\Sigma t : \alpha$. Moreover, we drop the Σ subscript when the context permits. Table 1 gives the typing rules: constant introduction, variable introduction (linear and non-linear),

Table 1: Typing rules for deriving typing judgments

$$\frac{}{\Gamma; \vdash_\Sigma c : \tau(c)}\ \text{(const.)}$$

$$\frac{}{\Gamma; x : \alpha \vdash_\Sigma x : \alpha}\ \text{(lin. var.)} \qquad \frac{}{\Gamma, x : \alpha; \vdash_\Sigma x : \alpha}\ \text{(var.)}$$

$$\frac{\Gamma; \Delta, x : \alpha \vdash_\Sigma t : \beta}{\Gamma; \Delta \vdash_\Sigma \lambda^o x.t : \alpha \multimap \beta}\ \text{(lin. abs.)} \qquad \frac{\Gamma; \Delta_1 \vdash_\Sigma t : \alpha \multimap \beta \quad \Gamma; \Delta_2 \vdash_\Sigma u : \alpha}{\Gamma; \Delta_1, \Delta_2 \vdash_\Sigma (t\,u) : \beta}\ \text{(lin. app.)}$$

$$\frac{\Gamma, x : \alpha; \Delta \vdash_\Sigma t : \beta}{\Gamma; \Delta \vdash_\Sigma \lambda x.t : \alpha \to \beta}\ \text{(abs.)} \qquad \frac{\Gamma; \Delta \vdash_\Sigma t : \alpha \to \beta \qquad \Gamma; \vdash_\Sigma u : \alpha}{\Gamma; \Delta \vdash_\Sigma (t\,u) : \beta}\ \text{(app.)}$$

linear abstraction and linear application, (non-linear) abstraction and (non-linear) application, in this order.

The fact that Γ is a set and Δ is a multi-set corresponds to implicitly allowing for the structural rules of contraction and weakening for the non-linear context, and disallowing them on the linear context.[7]

Remark 1. From a logical point of view, the theorems that can be proved using only the non-linear or the linar context are different. For instance, if $c : \alpha \to \alpha \to \beta$ and $d : \alpha \multimap \alpha \multimap \beta$ are constants of Σ, $x : \alpha; \vdash c\ x\ x : \beta$ is derivable as the following derivation shows:

$$\dfrac{\dfrac{\dfrac{}{x : \alpha; \vdash_\Sigma c : \alpha \to \alpha \to \beta}\ (\text{const.}) \qquad \dfrac{}{x : \alpha; \vdash_\Sigma x : \alpha}\ (\text{var.})}{x : \alpha; \vdash_\Sigma (c\ x) : \alpha \to \beta}\ (\text{app.}) \qquad \dfrac{}{x : \alpha; \vdash_\Sigma x : \alpha}\ (\text{var.})}{x : \alpha; \vdash_\Sigma ((c\ x)\ x) : \beta}\ (\text{app.})$$

whereas $; x : \alpha \vdash d\ x\ x : \beta$ is not (and $; x : \alpha, y : \alpha \vdash d\ x\ y : \beta$ is).

Remark 2. The linear context of the second premise in the non-linear application rule ($\Gamma; \vdash_\Sigma u : \alpha$) is empty. This is required in order to avoid duplicating or erasing linear variables by non-linear application to a linear variable. Otherwise we could have derivations such as:

$$\dfrac{\begin{array}{c}\vdots\\ ; \vdash_\Sigma \lambda x.c\ x\ x : \alpha \to \beta \end{array} \qquad ; y : \alpha \vdash_\Sigma y : \alpha}{; y : \alpha \vdash_\Sigma (\lambda x.c\ x\ x)\ y : \beta}\ (\text{app.})$$

Then we have that $y : \alpha$ belongs to the linear context, but $(\lambda x.c\ x\ x)\ y$ reduces to $c\ y\ y$ where y is duplicated.

Definition 5 (Linear and almost linear λ-terms)**.** A term without any λs, where each λ^o binds exactly one variable, and where no subterm contains more than one free occurrence of the same variable is a *linear* λ-term, otherwise it is *non-linear*.

A term where each λ^o binds exactly one variable, where each λ binds at least one variable, and no subterm contains more than one free occurrence of the same variable, except if the variable has an atomic type, is an *almost linear* λ-term.

The notion of linearity and almost linearity are important with respect to the tractability and the computational complexity of the

[7] Contraction corresponds to allowing the duplication of hypotheses, and weakening corresponds to allowing the deletion of useless hypothesis.

parsing algorithms, because they allow for characterising the set of almost linear λ-terms that are β-equivalent to some given almost linear term (Kanazawa 2017, p. 1120).

Definition 6 (Order)**.** The order $\mathsf{ord}(\tau)$ of a type $\tau \in \mathscr{T}(A)$ is inductively defined as:

- $\mathsf{ord}(a) = 1$ if $a \in A$
- $\mathsf{ord}(\alpha \multimap \beta) = \mathsf{ord}(\alpha \to \beta) = \max(1 + \mathsf{ord}(\alpha), \mathsf{ord}(\beta))$ otherwise

By extension, the order of a term is the order of its type.

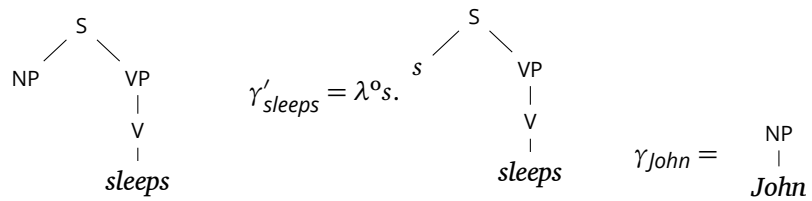
Remark 3. Second-order terms (i.e., terms whose order is 2) play an important role in our TAG encoding, and more generally for the expressive power of ACGs. A second-order term has type $a_1 \multimap a_2 \ldots \multimap a_n \multimap a$ where a, $a_1, \ldots, a_n$ are atomic types. If a signature Σ contains only first-order and second-order constants, an easy induction shows that ground terms (i.e., terms with no free variable) of atomic types in $\Lambda(\Sigma)$ do not contain any variable (bound or free) at all. In particular, they cannot have terms of the form $\lambda^{\mathrm{o}} x.u$ or $\lambda x.u$ as sub-terms.

We now assume the single atomic type T of trees and constants of this type (in Section 2.3 we make explicit how to systematically encode trees into λ-terms).

2.2.1 Substitution as function application

The ability for the tree of Figure 6(a) to accept a substitution at its NP node allows it to be considered as a function that takes a tree as argument and replaces the NP node by this argument. Hence we can represent it as the function γ'_{sleeps} shown in Figure 6(b) with $\gamma'_{sleeps} : T \multimap T$. A tree where no substitution can occur can be represented as $\gamma_{John} : T$ (see Figure 6(c)).

Figure 6: Functional interpretation of the substitution operation



(a) TAG initial tree (b) Function from trees to trees (c) Term of type T

Applying the function γ'_{sleeps} to the simple tree γ_{John} of Figure 6(c) and performing β-reduction gives the expected result as (1) shows.

(1) $\gamma'_{sleeps}\ \gamma_{John} = \left(\lambda^{o} s. \begin{array}{ccc} & \mathrm{S} & \\ s & & \mathrm{VP} \\ & & \mathrm{V} \\ & & \textit{sleeps} \end{array} \right) \begin{array}{c} \mathrm{NP} \\ \textit{John} \end{array} \rightarrow_{\beta} \begin{array}{ccc} & \mathrm{S} & \\ \mathrm{NP} & & \mathrm{VP} \\ \textit{John} & & \mathrm{V} \\ & & \textit{sleeps} \end{array}$

Note that despite the derived tree for *John sleeps* having a root labelled by S and the elementary tree for *John* having a root labelled by NP, they are represented by the terms $\gamma'_{sleeps}\ \gamma_{John}$ and γ_{John} which both have type T. The issue of recording the distinction is addressed in Section 4.3.

2.2.2 Adjunction as function application

In order to deal with the adjunction operation, we first observe what happens to the auxiliary tree $\beta_{seemingly}$ in Figure 4 (p. 534): a subtree of the tree it is adjoined to (the VP rooted subtree of α_{sleeps}) is substituted at the VP* foot node of $\beta_{seemingly}$. This means that the auxiliary tree also behaves as a function from trees to trees and can be represented as in Figure 7(a) with $\gamma'_{seemingly} : T \multimap T$. Then, a tree with an adjunction site can be represented by a term such as $\gamma''_{sleeps} : (T \multimap T) \multimap T$ in Figure 7(b). Note the higher-order type of γ''_{sleeps}.

In order to model the adjunction, we then apply γ''_{sleeps} to $\gamma'_{seemingly}$ and perform β-reductions as (2) shows.

$\gamma'_{seemingly} = \lambda^{o} v. \begin{array}{ccc} & \mathrm{VP} & \\ \mathrm{Adv} & & v \\ \textit{seemingly} & & \end{array}$

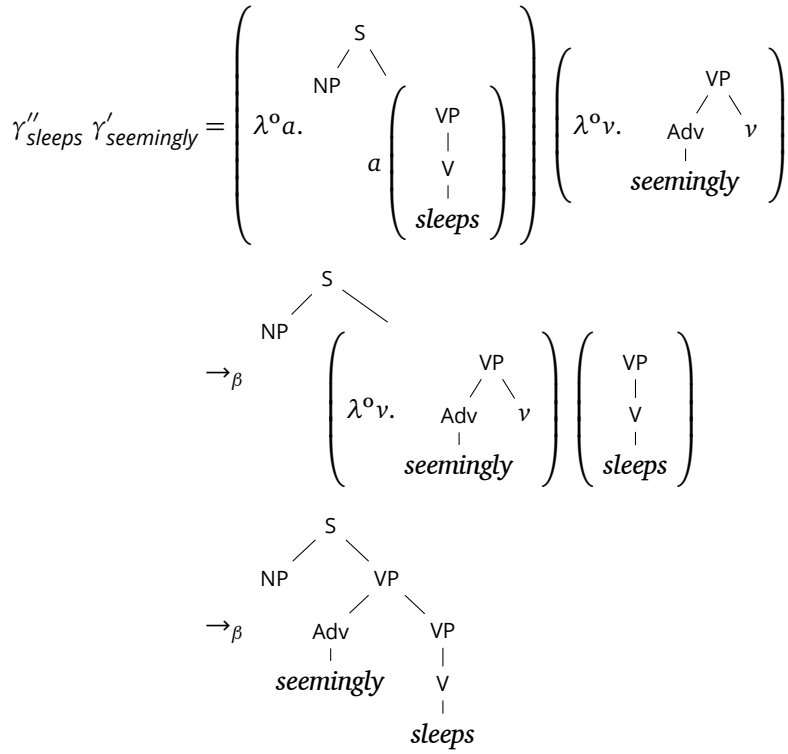
(a) Function from trees to trees

$\gamma''_{sleeps} = \lambda^{o} a. \begin{array}{ccc} & \mathrm{S} & \\ \mathrm{NP} & & \\ & a & \begin{array}{c} \mathrm{VP} \\ \mathrm{V} \\ \textit{sleeps} \end{array} \end{array}$

(b) Elementary tree ready to accept an adjunction

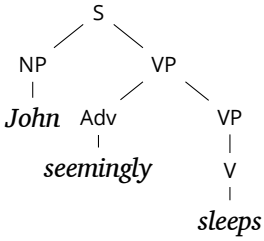
Figure 7: Functional interpretation of the adjunction operation

(2)



We are now (almost) in a position to define the function standing for the elementary tree representing the intransitive verb *sleeps* in its canonical form as in Figure 8 with $\gamma'''_{sleeps} : (T \multimap T) \multimap T \multimap T$. Such a term can be used to represent the TAG analysis of *John seemingly sleeps* shown in Figure 5 with the β-reduction of $\gamma'''_{sleeps}\ \gamma'_{seemingly}\ \gamma_{John}$ shown in (3).

(3) $\gamma'''_{sleeps}\ \gamma'_{seemingly}\ \gamma_{John} \rightarrow_{\beta}$



Remark 4 (No adjunction)**.** Typing γ'''_{sleeps} with $(T \multimap T) \multimap T \multimap T$ makes it require an adjunction (the first $(T \multimap T)$ argument) to return a

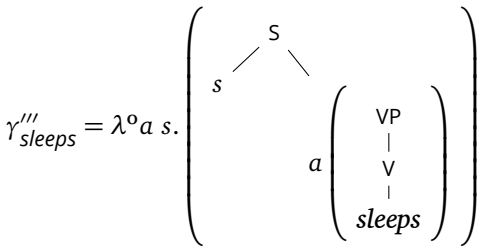
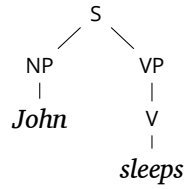


Figure 8:
Elementary tree representation available to substitution and adjunction operations

plain tree term of type T. But of course, we also want to use this term in case no adjunction in a TAG analysis would occur, as in *John sleeps*. We make use of a fake adjunction, applying γ'''_{sleeps} to the identity function $I = \lambda^{o} x.x : T \multimap T$.[8] Then (4) holds.

(4) $\gamma'''_{sleeps}\ I\ \gamma_{John} \rightarrow_{\beta}$



Finally, we also have to model the possible adjunction on the S node of α_{sleeps}. So the corresponding term γ_{sleeps} has type $(T \multimap T) \multimap (T \multimap T) \multimap T \multimap T$ where the first argument stands for the auxiliary tree to be adjoined at the S node, the second argument stands for the auxiliary tree to be adjoined at the VP node, and the third argument stands for the tree to be substituted at the NP node as Figure 9 shows.[9]

Remark 5 (Multiple adjunction)**.** Following Vijay-Shanker (1987), the typing we provide prevents two adjunctions from occurring at the same node in the same elementary tree. We discuss this difference with the multiple-adjunction approach of Schabes and Shieber (1994) in Section 5. Accordingly, an auxiliary tree should typically also allow for adjunction at its root. So instead of using $\gamma'_{seemingly} : T \multimap T$, we use the terms defined in Figure 10 in order to analyze sentences

[8] This idea is present in the seminal work using ACGs (de Groote 2002; Pogodalla 2004a), but also in the synchronous approach (Shieber 2004, 2006) and in Shieber (2014), in the notion of *vestigial auxiliary tree*.

[9] We could also allow adjunctions to the V node in a similar way. But we do not use examples of such adjunctions, and, for the sake of conciseness, we keep the type as small as required by the examples.

Figure 9: Encoding of α_{sleeps} available to substitution and adjunctions both at the VP and at the S nodes

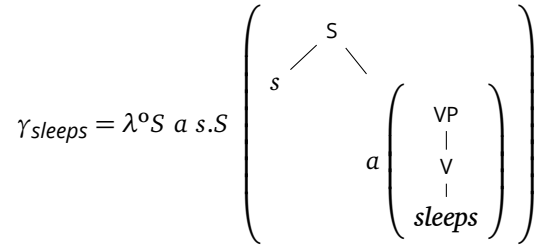


Figure 10: Auxiliary tree representation available to adjunction operations

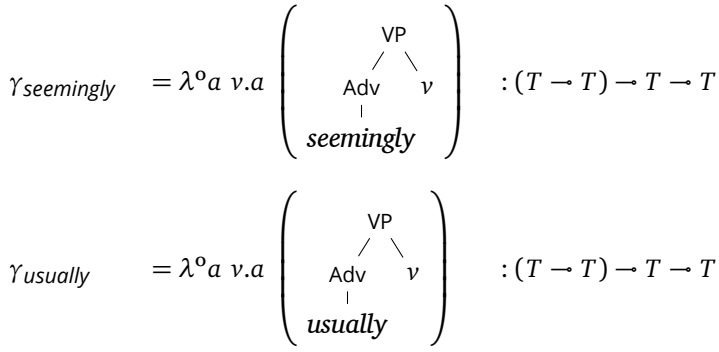
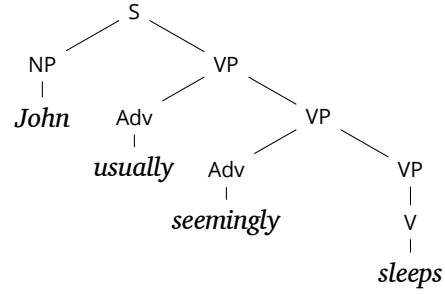


Figure 11: A TAG analysis of *John usually seemingly sleeps*



such as *John usually seemingly sleeps* as in Figure 11 with the term $\gamma_{sleeps}\ I\ (\gamma_{seemingly}\ (\gamma_{usually}\ I))\ \gamma_{John}$.[10]

2.3 *Trees and strings as λ-terms*

So far, we did not make explicit how to represent strings and trees as λ-terms. In particular, we did not explain how strings can combine and how the parent-child relation can be represented. While this is

[10] Although $\lambda^{o} v.\gamma'_{usually}\ (\gamma'_{seemingly}\ v) = \gamma_{seemingly}\ (\gamma_{usually}\ I)$, introducing $\gamma_{seemingly}$ and $\gamma_{usually}$ with this more complex types is important because, as we will see in Section 6, at the most abstract level, we want terms without any free or bound variable to represent derivations (see Remark 3).

quite standard, and because we use this encoding to implement the example grammars using the ACG toolkit, this section describes how it can be done.

2.3.1 Encoding strings

We encode strings over an alphabet C using the following higher-order signature $\Sigma^{C}_{\textit{strings}} = \langle A_\sigma, C, \tau_\sigma \rangle$ where:

- $A_\sigma = \{o\}$ contains a unique atomic type o;
- τ_σ is the constant function that maps any constant to the type σ, the string type, defined as $(o \multimap o)$. Note it is not an atomic type.

We use the notation $\triangleq$ to introduce terms or types that are defined using the atomic types and the constants of a signature, but are not atomic types nor constants of this signature. So $\sigma \triangleq (o \multimap o)$. We also define two other terms:

- $+ \triangleq \lambda^{o} f\ g . \lambda^{o} z . f\,(g\ z)$ (function composition, used with an infix notation) to represent *concatenation*;
- $\epsilon \triangleq \lambda^{o} x . x$ (the identity function) to represent the empty string.

It is easy to check that $+$ is associative and that ϵ is a neutral element for $+$.

Remark 6. If a and b are two strings, $a + b = \lambda^{o} z . a\ (b\ z)$. In this article, we usually do not unfold the definition of $+$ and we use the notation $x_1 + \ldots + x_n$ to represent the string $\lambda^{o} z . x_1\ (\ldots (x_n\ z) \ldots)$.

2.3.2 Encoding trees

Trees were originally defined in TAG using a mapping from positions (or Gorn addresses) to labels, elements of a vocabulary (Joshi *et al.* 1975). Hence, the same node label could be used to represent nodes of different arity. For instance, in Figure 2(a) (p. 533), the VP node has arity 1 whereas the VP node of Figure 2(c) has arity 2.

We prefer to represent trees as terms over a ranked alphabet as in Shieber (2004) and Comon *et al.* (2007) in order to make the encoding of trees as λ-terms easier. So we use the notation X_n with n the arity of the symbol X_n. It allows us to distinguish two nodes with n and m ($n \neq m$) children that would be mapped to the same label X by using the different symbols X_n and X_m. As terminal symbols can

only occur as leaves, they always have arity 0, so we do not use any subscript for them.

In order to encode the trees defined over a ranked alphabet $\mathscr{F}_a = (\mathscr{F}, arity)$, where *arity* is a mapping from $\mathscr{F}$ to $\mathbb{N}$, we use the following higher-order signature $\Sigma_{trees}^{\mathscr{F}_a} = \langle A_T, \mathscr{F}, \tau_T^{\mathscr{F}_a} \rangle$ where:

- $A_T = \{T\}$ contains a unique atomic type T, the type of trees;
- $\tau_T^{\mathscr{F}_a}$ is a function that maps any constant X such that $arity(X) = n$ to the type $\underbrace{T \multimap \cdots \multimap T}_{n \text{ times}} \multimap T$. If $arity(X) = 0$, then $\tau_T^{\mathscr{F}_a}(X) = T$.

For instance, the TAG elementary trees δ_{anchor}[11] of our running examples can be modelled as the functions (or terms) γ_{anchor} built on the signature Σ_{trees} as Table 2 shows.[12] Then (5) shows that

Table 2: Encoding of the TAG elementary trees with Σ_{trees}

	Terms of $\Lambda(\Sigma_{trees})$	Corresponding TAG tree
γ_{John}	$= \mathsf{NP}_1\ \mathit{John}$ $: T$	NP *John*
γ_{sleeps}	$= \lambda^{\circ} S\ a\ s.S\ (\mathsf{S}_2\ s\ (a\ (\mathsf{VP}_1\ (\mathsf{V}_1\ \mathit{sleeps}))))$ $: (T \multimap T) \multimap (T \multimap T) \multimap T \multimap T$	S NP VP V *sleeps*
$\gamma_{seemingly}$	$= \lambda^{\circ} a\ v.a\ (\mathsf{VP}_2\ (\mathsf{Adv}_1\ \mathit{seemingly})\ v)$ $: (T \multimap T) \multimap T \multimap T$	VP Adv VP* *seemingly*
$\gamma_{usually}$	$= \lambda^{\circ} a\ v.a\ (\mathsf{VP}_2\ (\mathsf{Adv}_1\ \mathit{usually})\ v)$ $: (T \multimap T) \multimap T \multimap T$	VP Adv VP* *usually*
γ_{hence}	$= \lambda^{\circ} a\ s.a\ (\mathsf{S}_2\ (\mathsf{Adv}_1\ \mathit{hence})\ s)$ $: (T \multimap T) \multimap T \multimap T$	S Adv S* *hence*
I	$= \lambda^{\circ} x.x : T \multimap T$	

[11] We use the notation δ_{anchor} to refer either to the initial tree α_{anchor} or to the auxiliary tree β_{anchor}.

[12] Note that *sleeps* and *seemingly* are used as constants of arity 0 and have type T. We also introduce an auxiliary tree that can adjoin to the S node.

$\gamma_5 = \gamma_{sleeps}\ I\ (\gamma_{seemingly}\ I)\ \gamma_{John}$ corresponds (modulo β) to the tree of Figure 5 (p. 535).

(5) $\gamma_5 = \gamma_{sleeps}\ I\ (\gamma_{seemingly}\ I)\ \gamma_{John}$
$\quad \rightarrow_\beta \mathsf{S}_2\ (\mathsf{NP}_1\ \textit{John})\ (\mathsf{VP}_2\ (\mathsf{Adv}_1\ \textit{seemingly})\ (\mathsf{VP}_1\ (\mathsf{V}_1\ \textit{sleeps})))$

We now want to relate the tree that is represented by the term $\gamma_{sleeps}\ I\ (\gamma_{seemingly}\ I)\ \gamma_{John} : T$ to the string *John seemingly sleeps* that is represented by the term $\textit{John} + \textit{seemingly} + \textit{sleeps} : \sigma$. We do this in Section 4, using an *interpretation* of the former as defined by an ACG.

3 ABSTRACT CATEGORIAL GRAMMARS

Grammars can be considered as a device to relate concrete objects to hidden underlying structures. For instance, context-free grammars relate strings to syntactic trees, and TAGs relate derived trees to derivation trees. However, in both cases, the underlying structure is not a first-class citizen of the formalism.

ACGs take another perspective and provide the user a direct way to define the parse structures of the grammar, the *abstract language*. Such structures are later on interpreted by a morphism, the *lexicon*, to get the concrete *object language*. The process of recovering an abstract structure from an object term is called *ACG parsing* and consists in inverting the lexicon. In this perspective, derivation trees of TAGs are represented as terms of an abstract language, while derived trees and yields are represented by terms of some other object languages: an object language of trees in the first case and an object language of strings in the second. We also use a logical language as the object language to express the semantic representations.

For the sake of self-containedness, we first review the definitions of de Groote (2001).

Definition 7 (Lexicon)**.** Let $\Sigma_1 = \langle A_1, C_1, \tau_1 \rangle$ and $\Sigma_2 = \langle A_2, C_2, \tau_2 \rangle$ be two higher-order signatures, Σ_1 being linear. A *lexicon* $\mathscr{L} = \langle F, G \rangle$ from Σ_1 to Σ_2 is such that:

- $F : A_1 \rightarrow \mathscr{T}(A_2)$. We also note $F : \mathscr{T}(A_1) \rightarrow \mathscr{T}(A_2)$, its homomorphic extension; that is, the function $\hat{F}$ that extends F such that $\hat{F}(\alpha \multimap \beta) = \hat{F}(\alpha) \multimap \hat{F}(\beta)$ and $\hat{F}(\alpha \rightarrow \beta) = \hat{F}(\alpha) \rightarrow \hat{F}(\beta)$;

- $G : C_1 \to \Lambda(\Sigma_2)$. We also note $G : \Lambda(\Sigma_1) \to \Lambda(\Sigma_2)$, its homomorphic extension; that is, the function $\hat{G}$ that extends G such that $\hat{G}(t\ u) = \hat{G}(t)\ \hat{G}(u)$, $\hat{G}(x) = x$, $\hat{G}(\lambda x.t) = \lambda x.\hat{G}(t)$, and $\hat{G}(\lambda^o x.t) = \lambda^o x.\hat{G}(t)$;
- F and G are such that for all $c \in C_1$, $\vdash_{\Sigma_2} G(c) : F(\tau_1(c))$ is provable.

We also use $\mathscr{L}$ instead of F or G.

The lexicon is the interpreting device of ACGs.

Definition 8 (Abstract Categorial Grammar and vocabulary)**.** An *abstract categorial grammar* is a quadruple $\mathscr{G} = \langle \Sigma_1, \Sigma_2, \mathscr{L}, \mathsf{S} \rangle$ where:

- $\Sigma_1 = \langle A_1, C_1, \tau_1 \rangle$ and $\Sigma_2 = \langle A_2, C_2, \tau_2 \rangle$ are two higher-order signatures. Σ_1 (resp. Σ_2) is called the *abstract vocabulary* (resp. the *object vocabulary*) and $\Lambda(\Sigma_1)$ (resp. $\Lambda(\Sigma_2)$) is the set of *abstract terms* (resp. the set of *object terms*).
- $\mathscr{L} : \Sigma_1 \to \Sigma_2$ is a lexicon and Σ_1 is a linear signature.
- $\mathsf{S} \in \mathscr{T}(A_1)$ is the *distinguished type* of the grammar.

Given an ACG $\mathscr{G}_{name} = \langle \Sigma_1, \Sigma_2, \mathscr{L}_{name}, \mathsf{S} \rangle$, instead of using $\mathscr{L}_{name}(\alpha) = \beta$ (resp. $\mathscr{L}_{name}(t) = u$) in order to express that the interpretation of the type α is the type β (resp. the interpretation of the term t is the term u), we use the following notational variants: $\mathscr{G}_{name}(\alpha) = \beta$ and $\alpha :=_{name} \beta$ (resp. $\mathscr{G}_{name}(t) = u$ and $t :=_{name} u$). The subscript may be omitted if clear from the context.

Definition 9 (Abstract and object languages)**.** Given an ACG $\mathscr{G}$, the *abstract language* is defined by

$$\mathscr{A}(\mathscr{G}) = \{t \in \Lambda(\Sigma_1) \mid \vdash_{\Sigma_1} t : \mathsf{S} \text{ is derivable}\}$$

The *object language* is defined by

$$\mathscr{O}(\mathscr{G}) = \{u \in \Lambda(\Sigma_2) \mid \exists t \in \mathscr{A}(\mathscr{G}) \text{ such that } u = \mathscr{L}(t)\}$$

In this article, we consider object languages such as strings or logical formulas, and abstract languages such as derivation trees. Some languages, such as the language of derived trees, will be considered sometimes as object languages, sometimes as abstract languages.

Parsing with an ACG $\mathscr{G}$ any term u that is built over the object vocabulary of $\mathscr{G}$ amounts to finding the abstract terms $t \in \mathscr{A}(\mathscr{G})$ such that $u = \mathscr{G}(t)$. In other words, ACG parsing is morphism inversion.

3.1 *ACG composition*

The lexicon defines the way structures are interpreted. It plays a crucial role in our proposal in two different ways. First, two interpretations may share the same abstract vocabulary, hence mapping a single structure into two different ones. For instance, the structure representing the derivations may be mapped both into a surface form and a semantic form. This composition is illustrated by $\mathcal{G}_{derived\ trees}$ and $\mathcal{G}_{sem.}$ sharing the $\Sigma_{derivations}$ vocabulary in Figure 12. It corresponds to the core model of the syntax-semantics interface as proposed in ACGs (de Groote 2001, Section 2.3), but also to the one proposed in synchronous TAG. It allows us to relate the derived trees and the semantic expressions that have the same derivation structures. We use this in Section 5 as our model of the syntax-semantics interface for TAG.

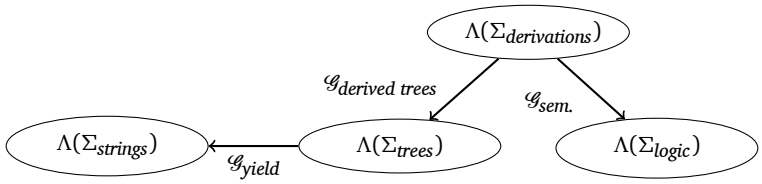


Figure 12: ACG composition modes

Second, the result of a first interpretation can itself be interpreted by a second lexicon when the object vocabulary of the first interpretation is the abstract vocabulary of the second one. This composition, illustrated by the $\mathcal{G}_{yield} \circ \mathcal{G}_{derived\ trees}$ composition in Figure 12, provides modularity. It also allows one to control admissible intermediate structures. For instance, the abstract language of $\mathcal{G}_{yield}$ may contain too many structures. If the object language of $\mathcal{G}_{derived\ trees}$ is a strict subset of this abstract language, then the object language of $\mathcal{G}_{yield} \circ \mathcal{G}_{derived\ trees}$ is a subset of the object language of $\mathcal{G}_{yield}$. We take advantage of this property in Section 4.3 to enforce the matching between node labels in substitution and adjunction operations, and to further restrict the set of derivations to only TAG derivations in Section 6. This ACG composition corresponds to the typical ACG way to control admissible parse structures.

3.2 *Formal properties of ACGs*

In this section, we review the main properties of ACGs and mention the relevant references. Two parameters are useful to define a hierarchy of ACGs: the *order* and the *complexity* of an ACG.

Definition 10 (Order and complexity of an ACG; ACG hierarchy)**.** The *order* of an ACG is the maximum of the orders of its abstract constants. The *complexity* of an ACG is the maximum of the orders of the realizations of its atomic types.

We call *second-order ACGs* the set of ACGs whose order is at most 2.

$\text{ACG}_{(n,m)}$ denotes the set of ACGs whose order is at most n and whose complexity is at most m.

For instance, in Figure 12, $\mathscr{G}_{yield}$ is a second-order ACG (because all the constants of Σ_{trees} are of type $T \multimap \cdots \multimap T$ with T atomic, hence are at most second-order). On the other hand, $\Sigma_{derivations}$ is third-order (it contains terms such as $c_{seemingly} : (\mathsf{VP} \multimap \mathsf{VP}) \multimap \mathsf{VP} \multimap \mathsf{VP}$, where NP and VP are atomic, that are third-order; see Section 4). Hence $\mathscr{G}_{derived\ trees}$ is third-order as well.

The class of second-order ACGs is of particular interest because of its polynomial parsing property (Salvati 2005). When considering strings as the object language, the generated languages coincide with multiple context-free languages (Salvati 2006). When considering trees, the generated languages coincide with the tree languages generated by hyperedge replacement grammars (Kanazawa 2009). A further refinement on the ACG hierarchy provides a fine-grained correspondence with regular (string or tree) languages, context-free string and linear context-free tree languages, or well-nested multiple context-free languages (string), in particular tree-adjoining languages. Table 3 (p. 549) sums up some of the formal properties of second-order ACGs (de Groote and Pogodalla 2004; Salvati 2006; Kanazawa and Salvati 2007; Kanazawa 2009).

For second-order ACGs, parsing algorithms and optimization techniques are grounded on well established fields such as type-theory and Datalog. Kanazawa (2007) showed how parsing of second-order ACGs reduces to Datalog querying, offering a general method for getting efficient tabular parsing algorithms (Kanazawa 2017). This parsing method applies whatever the object language: representing

	String language	Tree language
$ACG_{(1,n)}$	finite	finite
$ACG_{(2,1)}$	regular	regular
$ACG_{(2,2)}$	context-free	linear context-free
$ACG_{(2,3)}$	non-duplicating macro well-nested multiple context-free	$\subset$ 1-visit attribute grammar
$ACG_{(2,4)}$	mildly context-sensitive (multiple context-free)	tree-generating hyperedge replacement gram.
$ACG_{(2,4+n)}$	$ACG_{(2,4)}$	$ACG_{(2,4)}$

Table 3: The hierarchy of second-order ACGs

strings, trees, and also any kind of (almost linear) λ-terms. When the object language consists of logical formulas, the latter can then be parsed as well, and the resulting parse structures can further be interpreted (e.g., as strings) to implement surface realization (Kanazawa 2007). This also allows for deriving algorithms with specific properties such as prefix-correctness in a general way.[13]

The computational properties of lexicon inversion for ACGs have been studied for different classes of λ-terms.[14] It is worth noting that, as far as second-order ACGs are concerned, whatever the form of the semantic λ-term, lexicon inversion is decidable (Salvati 2010), even with replication and vacuous abstraction of variables, though with a high computational complexity. From a theoretical point of view, this corresponds to removing any kind of semantic monotonicity requirement for generation (Shieber 1988; Shieber *et al.* 1989) in a very general setting.[15]

The examples we present in this article use only almost linear semantic terms. This allows us to run them in the ACG toolkit. The latter implements the method of parsing by reduction to Datalog, and allows

[13] For a n^6 prefix-correct Earley recognizer for TAGs, see Kanazawa (2008b).

[14] See for instance Kanazawa (2017), Bourreau and Salvati (2011), and Bourreau (2012) for the linear, almost linear, and almost affine cases.

[15] In its strongest form, this requirement corresponds to having lexicalized semantic recipes (i.e., where at least one constant appears and no deletion is allowed). Linear and almost linear pure terms (i.e., where no constant occurs) are already dealt with in the Datalog reduction. Allowing deletion leads to more challenging issues. It is used, for instance, for the modelling of ellipsis (Kobele 2007; Bourreau 2012, 2013) or for providing intensional semantics to intension-insensitive words (de Groote and Kanazawa 2013).

us to parse strings, trees, and logical formulas using the grammars we propose. Large scale tests of the software are however ongoing work, and a quantitative evaluation is beyond the scope of this article.

Parsing with ACGs whose order is strictly greater than 2 is equivalent (Salvati 2005) to the decidability of the Multiplicative Exponential fragment of Linear Logic (MELL: Girard 1987).[16] De Groote (2015) shows a reduction of ACG parsing of higher-order ACGs to linear logic programming. It is of course trivially (by linearity) decidable for ACGs where the interpretation of abstract constants always introduces a constant of the object language. But even in this case, third-order ACGs can generate languages that are NP-complete (Yoshinaka and Kanazawa 2005; Salvati 2006). For higher-order ACGs, the ACG toolkit implements abstract term interpretation, but no parsing.

4 RELATING GENERALIZED DERIVATIONS, TAG DERIVED TREES, AND STRINGS WITH ABSTRACT CATEGORIAL GRAMMARS

From now on, we assume a TAG $\mathcal{G} = (\mathcal{I}, \mathcal{A})$ where $\mathcal{I}$ is the set of initial trees and $\mathcal{A}$ the set of auxiliary trees. The labels of the trees in $\mathcal{I} \cup \mathcal{A}$ range over the alphabet V^0, and $C \subset V^0$ is the terminal alphabet. V is the set of symbols of V^0 disambiguated by subscripting them with their arity (except for terminal symbols of arity 0), and $\mathcal{V}$ is the associated ranked alphabet.

4.1 *Derived trees and strings*

In the constructions of Section 2.3, we introduced two higher-order signatures: $\Sigma_{strings} = \Sigma^{C}_{strings}$ and $\Sigma_{trees} = \Sigma^{\mathcal{V}}_{trees}$. We can now relate terms built on them using an ACG $\mathcal{G}_{yield} = \langle \Sigma_{trees}, \Sigma_{strings}, \mathcal{L}_{yield}, T \rangle$ by specifying $\mathcal{L}_{yield}$ as follows:

- $\mathcal{L}_{yield}(T) = \sigma$ (a tree is interpreted as a string);
- for $X_n \in V \setminus C$, $\mathcal{L}_{yield}(X_n) = \lambda^{\circ} x_1 \ \ldots \ x_n . x_1 + \cdots + x_n$ (a tree labelled by a non-terminal symbol is interpreted by the function that concatenates the interpretation of its children);
- for $a \in C$, $\mathcal{L}_{yield}(a) = a$ (a terminal symbol is interpreted by the same symbol as a string).

[16] It has recently been proved to be decidable (Bimbó 2015).

For instance, (8) (p. 552) shows that the yield of the tree represented by $\gamma_5 = \gamma_{sleeps}\ I\ (\gamma_{seemingly}\ I)\ \gamma_{John}$ (p. 545) actually is $John + seemingly + sleeps$ (which can be rephrased as $\gamma_{sleeps}\ I\ (\gamma_{seemingly}\ I)\ \gamma_{John} :=_{yield} John + seemingly + sleeps$). Indeed, we have (6).

(6)
$$\begin{aligned}
\mathcal{G}_{yield}(\gamma_{John}) &= \mathcal{G}_{yield}(\mathsf{NP}_1\ John) \\
&= \mathcal{G}_{yield}(\mathsf{NP}_1)\ \mathcal{G}_{yield}(John) \\
&\qquad\qquad \text{because } \mathcal{G}_{yield} \text{ is a morphism} \\
&= (\lambda^{o} x.x)\ John \\
&\qquad\qquad \text{by definition of } \mathcal{G}_{yield} \text{ on constants} \\
&\rightarrow_{\beta} John
\end{aligned}$$

And with $\gamma_7 = \mathsf{VP}_2\ (\mathsf{Adv}_1\ seemingly)\ (\mathsf{VP}_1\ (\mathsf{V}_1\ sleeps))$, we have (7),

(7)
$$\begin{aligned}
\mathcal{G}_{yield}(\gamma_7) &= \mathcal{G}_{yield}(\mathsf{VP}_2\ (\mathsf{Adv}_1\ seemingly)\ (\mathsf{VP}_1\ (\mathsf{V}_1\ sleeps))) \\
&= \mathcal{G}_{yield}(\mathsf{VP}_2)\ (\mathcal{G}_{yield}(\mathsf{Adv}_1\ seemingly))\ (\mathcal{G}_{yield}(\mathsf{VP}_1\ (\mathsf{V}_1\ sleeps))) \\
&\qquad\qquad \text{because } \mathcal{G}_{yield} \text{ is a morphism} \\
&= (\lambda^{o} x_1\ x_2.x_1 + x_2) \\
&\qquad (\mathcal{G}_{yield}(\mathsf{Adv}_1\ seemingly))\ (\mathcal{G}_{yield}(\mathsf{VP}_1\ (\mathsf{V}_1\ sleeps))) \\
&\qquad\qquad \text{by definition of } \mathcal{G}_{yield} \text{ on } \mathsf{VP}_2 \\
&\rightarrow_{\beta} \mathcal{G}_{yield}(\mathsf{Adv}_1\ seemingly) + \mathcal{G}_{yield}(\mathsf{VP}_1\ (\mathsf{V}_1\ sleeps)) \\
&= (\mathcal{G}_{yield}(\mathsf{Adv}_1)\ \mathcal{G}_{yield}(seemingly)) \\
&\qquad + (\mathcal{G}_{yield}(\mathsf{VP}_1)\ (\mathcal{G}_{yield}(\mathsf{V}_1\ sleeps))) \\
&\qquad\qquad \text{because } \mathcal{G}_{yield} \text{ is a morphism} \\
&= ((\lambda^{o} x.x)\ \mathcal{G}_{yield}(seemingly)) + ((\lambda^{o} x.x)\ (\mathcal{G}_{yield}(\mathsf{V}_1\ sleeps))) \\
&\qquad\qquad \text{by definition of } \mathcal{G}_{yield} \text{ on } \mathsf{Adv}_1 \text{ and } \mathsf{VP}_1 \\
&= \mathcal{G}_{yield}(seemingly) + \mathcal{G}_{yield}(\mathsf{V}_1\ sleeps) \\
&= seemingly + (\mathcal{G}_{yield}(\mathsf{V}_1)\ (\mathcal{G}_{yield}(sleeps))) \\
&= seemingly + ((\lambda^{o} x.x)\ sleeps) \\
&= seemingly + sleeps
\end{aligned}$$

hence (8):

(8)

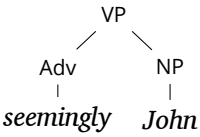
$$\begin{aligned}
\mathscr{G}_{yield}(\gamma_5) &= \mathscr{G}_{yield}(\mathsf{S}_2\ (\mathsf{NP}_1\ \textit{John})\ (\mathsf{VP}_2\ (\mathsf{Adv}_1\ \textit{seemingly})\ (\mathsf{VP}_1\ (\mathsf{V}_1\ \textit{sleeps})))) \\
&\qquad\qquad \text{by (5), p. 545} \\
&= \mathscr{G}_{yield}(\mathsf{S}_2)\ (\mathscr{G}_{yield}(\mathsf{NP}_1\ \textit{John}))\ (\mathscr{G}_{yield}(\gamma_7)) \\
&= (\lambda^{\mathrm{o}} x_1\ x_2 . x_1 + x_2)\ \textit{John}\ (\textit{seemingly} + \textit{sleeps}) \\
&\qquad\qquad \text{by (6) and (7)} \\
&\rightarrow_\beta \textit{John} + \textit{seemingly} + \textit{sleeps}
\end{aligned}$$

4.2 *Derivation trees and derived trees*

In this section, we illustrate how to introduce more control on the accepted structures. Note indeed that according to the definition of $\mathscr{G}_{yield}$, whatever is a closed term of type T belongs to its abstract language. For instance, $\gamma_{13} = \gamma_{seemingly}\ I\ \gamma_{John}$ is a well-typed term of type T corresponding to the tree of Figure 13 as (9) shows. Consequently, its interpretation $\textit{seemingly} + \textit{John}$ belongs to the object language.

(9)
$$\begin{aligned}
\gamma_{13} &= \gamma_{seemingly}\ I\ \gamma_{John} \\
&\rightarrow_\beta \mathsf{VP}_2\ (\mathsf{Adv}_1\ \textit{seemingly})\ (\mathsf{NP}_1\ \textit{John}) \\
&\coloneqq_{yield} \textit{seemingly} + \textit{John}
\end{aligned}$$

Figure 13:
Tree in the abstract language of $\mathscr{G}_{yield}$



In order to avoid such terms belonging to the language we are interested in, we provide another ACG, $\mathscr{G}_{derived\ trees}$, such that its object language is a strict subset of $\mathscr{A}(\mathscr{G}_{yield})$ (see Figure 12 p. 547). Consequently, the object language of $\mathscr{G}_{yield} \circ \mathscr{G}_{derived\ trees}$ is a subset (strict in this case, as expected) of $\mathscr{O}(\mathscr{G}_{yield})$. $\mathscr{G}_{derived\ trees}$ is defined as $\mathscr{G}_{derived\ trees} = \langle \Sigma_{derivations}, \Sigma_{trees}, \mathscr{L}_{derived\ trees}, \mathsf{S} \rangle$.

4.3 *Generalized derivation trees*

4.3.1 A vocabulary for derivations: the $\Sigma_{derivations}$ signature

Adjoining $\gamma_{seemingly}$ on γ_{John} is possible in $\Lambda(\Sigma_{trees})$ because the type T does not take the node labels into account. Hence, there is, for in-

stance, no distinction between trees rooted by VP and trees rooted by NP. We introduce this distinction in a new signature $\Sigma_{derivations} = \langle A_{derivations}, C_{derivations}, \tau_{derivations} \rangle$. $A_{derivations} = V^0$ is the set of non-terminal symbols of the TAG grammar $\mathcal{G}$. Then, for any $\delta_{anchor} \in \mathcal{I} \cup \mathcal{A}$ an elementary tree of $\mathcal{G}$, we define c_{anchor} a constant of type $(X^1 \multimap X^1) \multimap \cdots \multimap (X^n \multimap X^n) \multimap Y^1 \multimap \cdots \multimap Y^m \multimap \alpha$ where:

- the X^i are the labels of the n internal nodes of δ_{anchor} labelled with a non-terminal where an adjunction is possible (by convention we use the breadth-first traversal); [17]
- the Y^i are the labels of the m leaves labelled with non-terminals, *not counting the foot node if δ_{anchor} is an auxiliary tree*, of δ_{anchor} (by convention, we use the left-right order);
- let Z be the label of the root node of δ_{anchor}. $\alpha = Z$ if $\delta_{anchor} \in \mathcal{I}$ is an initial tree, and $\alpha = Z'' \multimap Z'$ with Z'' corresponding to the label of the foot node and Z' corresponding to the label of the root node if $\delta_{anchor} \in \mathcal{A}$ is an auxiliary tree.[18] In the latter case, we call $Z'' \multimap Z'$ the *modifier type* of the constant modelling the auxiliary tree.

We get for instance the constants typed as in (10)[19] from the elementary trees of Figure 2 (p. 533).

(10)
$$c_{sleeps} : (\mathsf{S} \multimap \mathsf{S}) \multimap (\mathsf{VP} \multimap \mathsf{VP}) \multimap \mathsf{NP} \multimap \mathsf{S}$$
$$c_{John} : \mathsf{NP}$$
$$c_{seemingly} : (\mathsf{VP} \multimap \mathsf{VP}) \multimap \mathsf{VP} \multimap \mathsf{VP}$$

For each non-terminal X of the TAG grammar where an adjunction can occur, we also define $I_X : X \multimap X$ as in (11). These constants play a similar role as I at the Σ_{trees} level: they are used when a TAG derivation does not involve any adjunction on sites where it would be possible to have some.

[17] Instead of the types $(X^i \multimap X^i)$, we may have types $X^i_{i_1} \multimap X^i_{i_2}$ to denote a difference between the top and the bottom feature of the node of label X^i. This is in particular used to account for selecting adjoining constraints as described in Feature-based TAG (FTAG: Vijay-Shanker and Joshi 1988, 1991). See note 18.

[18] In standard TAG, we typically have $Z = Z' = Z''$. However, we shall see examples in Sections 5.3.2 and 7 where the distinction between Z, Z', and Z'' is relevant.

[19] We assume that no adjunction is allowed on the V node nor on the Adv node.

(11)
$$I_{\mathsf{S}} : \mathsf{S} \multimap \mathsf{S}$$
$$I_{\mathsf{VP}} : \mathsf{VP} \multimap \mathsf{VP}$$

Then the set of typed constants of $\Sigma_{derivations}$ is $C_{derivations} = \{c_{anchor} | \ \delta_{anchor} \in \mathcal{I} \cup \mathcal{A}\} \cup \{I_X | \ X \in V^0\}$ and $\tau_{derivations}$ is the associated typing function defined as above. The typing provided by $\Sigma_{derivations}$ now disallows the application of $c_{seemingly}\ I_{\mathsf{VP}} : \mathsf{VP} \multimap \mathsf{VP}$ to $c_{John} : \mathsf{NP}$.

We now need to relate the terms of $\Lambda(\Sigma_{derivations})$ to the terms of $\Lambda(\Sigma_{trees})$ by a suitable interpretation.

4.3.2 Interpretation of derivations as derived trees: the $\mathcal{G}_{derived\ trees}$ ACG

In order to define $\mathcal{G}_{derived\ trees} = \langle \Sigma_{derivations}, \Sigma_{trees}, \mathcal{L}_{derived\ trees}, \mathsf{S} \rangle$ we are left with defining $\mathcal{L}_{derived\ trees}$. All the atomic types (S, VP, etc.) are interpreted as trees (i.e., with the T type). And for a TAG elementary tree δ_{anchor}, the constant c_{anchor} is interpreted by γ_{anchor} (defined in Section 2.3.2). This leads us to the interpretations of Table 4.

Table 4: Interpretation of $\Sigma_{derivations}$ constants by $\mathcal{G}_{derived\ trees}$

c_{John}	$: \mathsf{NP}$	
	$:=_{derived\ trees}\ \gamma_{John}$	$= \mathsf{NP}_1\ \textit{John} : T$
c_{sleeps}	$: (\mathsf{S} \multimap \mathsf{S}) \multimap (\mathsf{VP} \multimap \mathsf{VP}) \multimap \mathsf{NP} \multimap \mathsf{S}$	
	$:=_{derived\ trees}\ \gamma_{sleeps}$	$= \lambda^{\mathrm{o}} S\ a\ s.S\ (\mathsf{S}_2\ s\ (a\ (\mathsf{VP}_1\ (\mathsf{V}_1\ \textit{sleeps}))))$ $: (T \multimap T) \multimap (T \multimap T) \multimap T \multimap T$
$c_{seemingly}$	$: (\mathsf{VP} \multimap \mathsf{VP}) \multimap \mathsf{VP} \multimap \mathsf{VP}$	
	$:=_{derived\ trees}\ \gamma_{seemingly}$	$= \lambda^{\mathrm{o}} a\ v.a\ (\mathsf{VP}_2\ (\mathsf{Adv}_1\ \textit{seemingly})\ v)$ $: (T \multimap T) \multimap T \multimap T$
$c_{usually}$	$: (\mathsf{VP} \multimap \mathsf{VP}) \multimap \mathsf{VP} \multimap \mathsf{VP}$	
	$:=_{derived\ trees}\ \gamma_{usually}$	$= \lambda^{\mathrm{o}} a\ v.a\ (\mathsf{VP}_2\ (\mathsf{Adv}_1\ \textit{usually})\ v)$ $: (T \multimap T) \multimap T \multimap T$
c_{hence}	$: (\mathsf{S} \multimap \mathsf{S}) \multimap \mathsf{S} \multimap \mathsf{S}$	
	$:=_{derived\ trees}\ \gamma_{hence}$	$= \lambda^{\mathrm{o}} a\ s.a\ (\mathsf{S}_2\ (\mathsf{Adv}_1\ \textit{hence})\ s)$ $: (T \multimap T) \multimap T \multimap T$
I_{S}	$: \mathsf{S} \multimap \mathsf{S}$	
	$:=_{derived\ trees}\ I$	$= \lambda^{\mathrm{o}} x.x : T \multimap T$
I_{VP}	$: \mathsf{VP} \multimap \mathsf{VP}$	
	$:=_{derived\ trees}\ I$	$= \lambda^{\mathrm{o}} x.x : T \multimap T$

In Section 4.2, we noticed that $\gamma_{13} = \gamma_{seemingly}\ I\ \gamma_{John} : T \in \mathcal{A}(\mathcal{G}_{yield})$ (see Equation (9) on page 552). By definition of the object language of an ACG, its interpretation $\mathcal{G}_{yield}(\gamma_{13}) = \textit{seemingly} + \textit{John}$ is such that $\mathcal{G}_{yield}(\gamma_{13}) \in \mathcal{O}(\mathcal{G}_{yield})$.

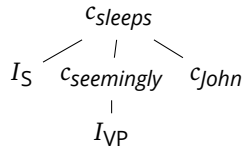
However, $\gamma_{13} \notin \mathcal{O}(\mathcal{G}_{\textit{derived trees}})$. Indeed, there is no c_{13} such that $\mathcal{G}_{\textit{derived trees}}(c_{13}) = \gamma_{13}$. A simple argument using the linearity of the interpretation shows that only $c_{\textit{seemingly}}$ (once and only once), $c_{\textit{John}}$ (once and only once), and I_X can be used. But $c_{\textit{John}}$ can not combine with any of the other terms (none of them use the type NP). Consequently, $\textit{seemingly}+\textit{John} \notin \mathcal{O}(\mathcal{G}_{\textit{yield}} \circ \mathcal{G}_{\textit{derived trees}})$, as is expected from the TAG grammar.

4.3.3 $\mathcal{G}_{\textit{derived trees}}$ abstract terms and generalized derivation trees

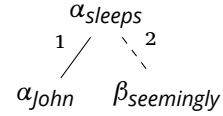
It is interesting to note that abstract terms of $\mathcal{G}_{\textit{derived trees}}$ describe the way the encoding of trees in $\Sigma_{\textit{trees}}$ can combine. We can see this combination in terms such as $\gamma_5 = \gamma_{\textit{sleeps}}\ I\ (\gamma_{\textit{seemingly}}\ I)\ \gamma_{\textit{John}}$, but it is in some sense an artifact of the definition we gave: γ_5 β-reduces to a tree that does not show this structure anymore. However, a term such as $c_5 = c_{\textit{sleeps}}\ I_{\mathsf{S}}\ (c_{\textit{seemingly}}\ I_{\mathsf{VP}})\ c_{\textit{John}}$ does not further β-reduce. Because we considered substitution as function application on arguments of atomic types and adjunction as function application on arguments of second-order types, c_5 keeps track of the adjunction of I_{S} on $c_{\textit{sleeps}}$, of the adjunction of I_{VP} on $c_{\textit{seemingly}}$, of the adjunction of the latter result on $c_{\textit{sleeps}}$, and of the substitution of $c_{\textit{John}}$. And the relation $\mathcal{G}_{\textit{derived trees}}(c_5) = \gamma_5$ expresses the relation between the record of these operations and the resulting derived tree.

We can represent c_5 as a tree (see Figure 14(a)): each node corresponds to a constant, applied to the terms represented by the children of the node. It makes explicit how similar to TAG derivation trees they are (Figure 14(b)). There is a one to one correspondence despite the following superficial differences:

- in the abstract term representation, the fake adjunctions (of I_X) are represented;



(a) Abstract term of $\mathcal{G}_{\textit{derived trees}}$



(b) TAG derivation tree

Figure 14: Derivation representations

- instead of specifying the role of the arguments with the Gorn address, we set a specific order for the arguments.

All the objects of a TAG grammar now have an ACG counterpart:

- terms of the abstract language of $\mathscr{G}_{\textit{derived trees}}$ correspond to the TAG derivation trees;[20]
- terms of $\Lambda(\Sigma_{\textit{trees}})$ that are in the object language of $\mathscr{G}_{\textit{derived trees}}$ correspond to the TAG derived trees;
- terms of $\Lambda(\Sigma_{\textit{strings}})$ that are in the object language of $\mathscr{G}_{\textit{yield}} \circ \mathscr{G}_{\textit{derived trees}}$ correspond to the TAG generated language.

(12) and (13) illustrate these correspondences for the abstract term $c_5 = c_{\textit{sleeps}}\ I_{\mathsf{S}}\ (c_{\textit{seemingly}}\ I_{\mathsf{VP}})\ c_{\textit{John}}$ representing the derivation for the analysis of *John seemingly sleeps*.

(12)
$$\begin{aligned}&\mathscr{G}_{\textit{derived trees}}(c_{\textit{sleeps}}\ I_{\mathsf{S}}\ (c_{\textit{seemingly}}\ I_{\mathsf{VP}})\ c_{\textit{John}})\\ &= \gamma_{\textit{sleeps}}\ I\ (\gamma_{\textit{seemingly}}\ I)\ \gamma_{\textit{John}}\\ &= \mathsf{S}_2\ (\mathsf{NP}_1\ \textit{John})\ (\mathsf{VP}_2\ (\mathsf{Adv}_1\ \textit{seemingly})\ (\mathsf{VP}_1\ (\mathsf{V}_1\ \textit{sleeps})))\\ &\quad \text{by (5), p. 545}\end{aligned}$$

(13)
$$\begin{aligned}&\mathscr{G}_{\textit{yield}} \circ \mathscr{G}_{\textit{derived trees}}(c_{\textit{sleeps}}\ I_{\mathsf{S}}\ (c_{\textit{seemingly}}\ I_{\mathsf{VP}})\ c_{\textit{John}})\\ &= \textit{John} + (\textit{seemingly} + \textit{sleeps}) \quad \text{by (12) and (8) (p. 552)}\end{aligned}$$

Remark 7 ($\mathscr{G}_{\textit{derived trees}}$ terms and description of trees)**.** Let us have a look at the $(X \multimap X)$ type of the argument of an abstract constant of $\mathscr{G}_{\textit{derived trees}}$ and at its interpretation. In $c_{\textit{sleeps}}$ for instance, the argument with type $(\mathsf{VP} \multimap \mathsf{VP})$ is interpreted by the a variable of $\gamma_{\textit{sleeps}}$ (see Table 4 on page 554). The position of a in the term $S\ (\mathsf{S}_2\ s\ (a\ (\mathsf{VP}_1\ (\mathsf{V}_1\ \textit{sleeps}))))$ makes it explicit that the result of a applied to $(\mathsf{VP}_1\ (\mathsf{V}_1\ \textit{sleeps}))$, hence the latter term itself, is dominated by the second child of S_2 (the variable s being the first one). So, in some sense, the type $(\mathsf{VP} \multimap \mathsf{VP})$ of a corresponds to the *dominance constraint* between the node where a occurs (here the second child of S_2) and the root node of its argument (here $\mathsf{VP}_1\ (\mathsf{V}_1\ \textit{sleeps})$), as in the tree descriptions of Vijay-Shanker (1992): the root node of the argument of a is always dominated by the node where a occurs. In particular, replacing a by $\lambda^{\circ} x.x$ corresponds to having these two nodes identified.

[20] There remain some reservations that Section 6 clears up, though. See Remark 8.

Remark 8 (Generalized derivations and TAG derivation trees)**.** It should be noted that $\mathscr{G}_{derived\ trees}$ and $\mathscr{G}_{yield} \circ \mathscr{G}_{derived\ trees}$ are *not* second-order ACGs. It means that the polynomial parsing results do not directly apply. But we know that TAG parsing is polynomial. So what is happening here?

The answer is that while $\mathscr{G}_{derived\ trees}$ constrains the string language more than $\mathscr{G}_{yield}$ does, it does not constrain it enough to generate only the corresponding TAG language. There is indeed no difference between the type of the encoding of an elementary tree of root S with a substitution node S and the encoding of an auxiliary tree with root and foot nodes labelled by S: it is $\mathsf{S} \multimap \mathsf{S}$ in both cases.

The solution, that we develop in Section 6, is to further control $\mathscr{A}(\mathscr{G}_{derived\ trees})$ with another ACG $\mathscr{G}_{TAG}$ such that $\mathscr{O}(\mathscr{G}_{TAG}) \subset \mathscr{A}(\mathscr{G}_{derived\ trees})$ just as $\mathscr{G}_{derived\ trees}$ allows us to control $\mathscr{A}(\mathscr{G}_{yield})$. The general architecture is then the one that Figure 15 describes.

But while this additional control is necessary to have a faithful encoding of TAG, it is not necessary to provide the semantic interpretation of the derivation trees (and may somewhat obfuscate it). That is why we first present the syntax-semantic interface we propose and delay the final encoding of TAG (that corresponds to the one of de Groote 2002) to Section 6.

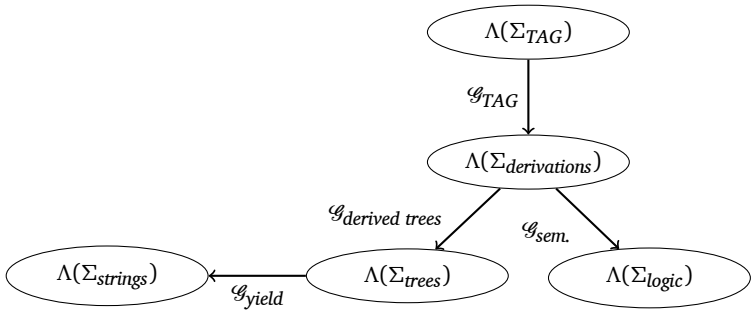


Figure 15: ACG composition for control and syntax-semantics interface

5 SEMANTIC CONSTRUCTION

In the previous section, we defined a signature $\Sigma_{derivations}$ to represent derivation structures as terms of $\Lambda(\Sigma_{derivations})$. We now use this signature as a pivot to transfer these structures into semantic representations. From a practical point of view, as mentioned in Section 3.1, it

amounts to defining an ACG $\mathcal{G}_{sem.} = \langle \Sigma_{derivations}, \Sigma_{logic}, \mathcal{L}_{sem.}, \mathsf{S} \rangle$ and composing it with $\mathcal{G}_{derived\ trees}$ thanks to the shared abstract vocabulary $\Sigma_{derivations}$. The object vocabulary Σ_{logic} of this ACG is the vocabulary for defining the semantic representations. In this article, we use higher-order logic (and, more often than not, simply first-order logic). Other languages, such as description languages to express underspeficied representations (Bos 1995; Egg *et al.* 2001), modal logic languages, etc. are possible as well. But we want to focus on *how* to build semantic representations rather than on the semantic modelling of some linguistic phenomenon itself.

5.1 *A vocabulary for semantic representations:* Σ_{logic}

We first define the object vocabulary $\Sigma_{logic} = \langle A_{logic}, C_{logic}, \tau_{logic} \rangle$ as in Table 5 with $A_{logic} = \{e, t\}$ the atomic types for *entities* and *truth values* respectively. As usual, we write the λ-term $\exists(\lambda x.P)$ as $\exists x.P$. The same, *mutatis mutandis*, holds for $\forall$. Note that, in this signature, we also use the non-linear implication, as a lot of semantic formulas (e.g., adjectives and quantifiers) use non linearity of entities. But we stay within the fragment of almost linear terms as only terms of atomic type are duplicated (see Definition 5 on page 537).[21]

Table 5: The vocabulary Σ_{logic}

Logical constants

$\wedge$	$: t \multimap t \multimap t$	$\vee : t \multimap t \multimap t$	
$\Rightarrow$	$: t \multimap t \multimap t$	$\neg : t \multimap t$	
$\exists$	$: (e \to t) \multimap t$	$\forall : (e \to t) \multimap t$	

Non-logical constants

john	$: e$	**love, chase**	$: e \multimap e \multimap t$
sleep	$: e \multimap t$	**seemingly, usually, hence**	$: t \multimap t$
seem	$: e \multimap (e \multimap t) \multimap t$	**claim, think**	$: e \multimap t \multimap t$
WHO	$: (e \multimap t) \multimap t$	**big, black, dog, cat**	$: e \multimap t$

5.2 *Generalized derivation-based interpretation*

The first step in defining $\mathcal{G}_{sem.}$, to interpret the abstract vocabulary $\Sigma_{derivations}$ into types and terms built on the object vocabulary Σ_{logic},

[21] For the sake of simplicity, we use simplified extensional types e and t. A more accurate semantic would require, for instance, intensional types.

Table 6: Interpretation by $\mathscr{G}_{sem.}$ of the $\Sigma_{derivations}$ vocabulary

$\mathsf{S} :=_{sem.} t$	$\mathsf{NP} :=_{sem.} (e \to t) \multimap t$	$\mathsf{N} :=_{sem.} e \to t$
$\mathsf{VP} :=_{sem.} e \to t$	$\mathsf{WH} :=_{sem.} (e \to t) \multimap t$	

(a) Interpretation of the atomic types

c_{John}	$:=_{sem.} \lambda^{o} P.P$ **john**
c_{sleeps}	$:=_{sem.} \lambda^{o} adv_{\mathsf{S}}\ adv_{\mathsf{VP}}\ subj.adv_{\mathsf{S}}\ (subj\ (adv_{\mathsf{VP}}\ (\lambda x.$**sleep** $x)))$
$c_{seemingly}$	$:=_{sem.} \lambda^{o} adv_{mod}\ pred.adv_{mod}\ (\lambda x.$**seemingly** $(pred\ x))$
$c_{usually}$	$:=_{sem.} \lambda^{o} adv_{mod}\ pred.adv_{mod}\ (\lambda x.$**usually** $(pred\ x))$
c_{hence}	$:=_{sem.} \lambda^{o} adv_{mod}\ pred.adv_{mod}\ ($**hence** $pred)$
I_{S}	$:=_{sem.} \lambda^{o} x.x$
I_{VP}	$:=_{sem.} \lambda^{o} x.x$

(b) Interpretation of the constants

is to define the interpretation of the atomic types (S, VP…). We simply follow the standard interpretation of these syntactic types into the semantic types as proposed by Montague (1973). This results in the interpretation described in Table 6(a). The interpretation of the constants follows, as in Table 6(b).[22] We do not repeat here the type of the constants c_{anchor} of $\Sigma_{derivations}$, nor the constraint that the image of this type has to be the type of $\mathscr{G}_{sem.}(c_{anchor})$ (e.g., the type of c_{John} is NP, hence $\mathscr{G}_{sem.}(c_{John}) : \mathscr{G}_{sem.}(\mathsf{NP}) = (e \to t) \multimap t$). But the reader can check that this proviso holds.

We let the reader check that for our favourite example, $c_5 :=_{sem.}$ **seemingly** (**sleep john**) as (14) shows.

(14)
$$\begin{aligned} c_5 &= c_{sleeps}\ I_{\mathsf{S}}\ (c_{seemingly}\ I_{\mathsf{VP}})\ c_{John} \\ &:=_{sem.} \textbf{seemingly}\ (\textbf{sleep john}) \end{aligned}$$

5.3 *From derivation dependencies to semantic dependencies*

We now turn to accounting for the mismatch between the dependencies as expressed in the derivation trees and in the logical semantic representations.

[22] The types now also use the intuitionistic implication →. This is required when variables that are abstracted over appear more than once in the semantic recipes. This is in particular the case for entities in quantified formulas, or in the semantics of intersective adjectives (see next section).

5.3.1 Long-distance dependencies

The first mismatch we consider, in order to make explicit what exactly this mismatch refers to, relates to the classical examples (15–17).

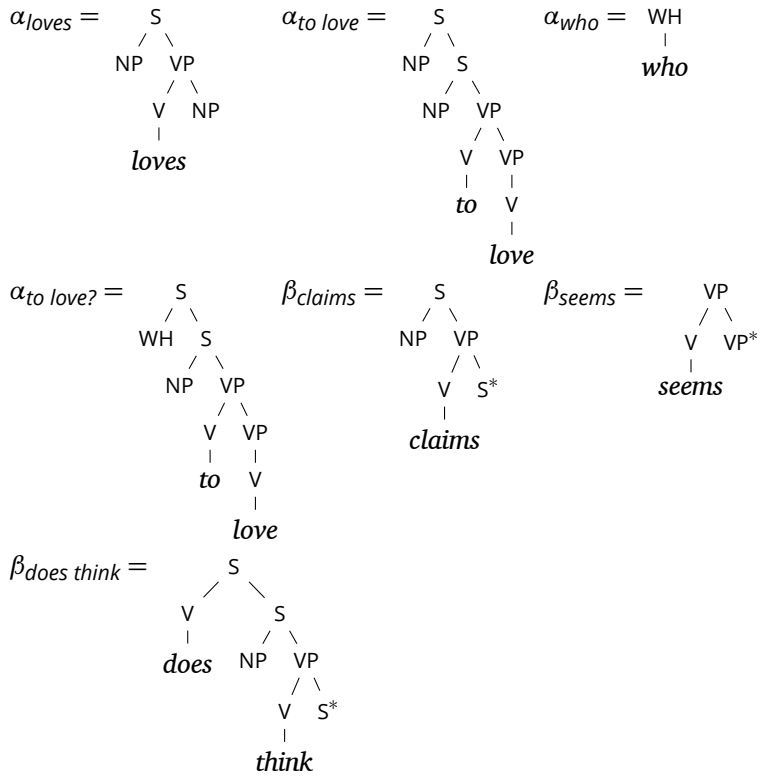
(15) Paul claims John loves Mary.

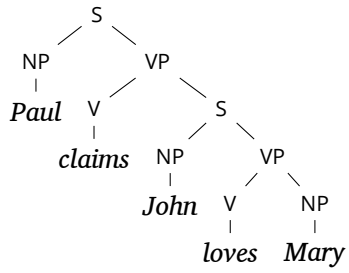
(16) Mary, Paul claims John seems to love.

(17) Who does Peter think Paul claims John seems to love?

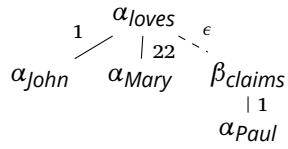
The TAG analysis relies on the elementary trees of Figure 16 and results in the derived tree and derivation tree of Figure 17 (p. 561) for (15). The mismatch appears in the contrast between the derivation tree where α_{loves} scopes over β_{claims} whereas the opposite scoping is to be expected from a semantic point of view. A similar effect occurs with (16) as the derivation tree, shown in Figure 18(b) (p. 561), makes $\alpha_{to\ love}$ scope over both β_{claims} and β_{seems}, while semantically both

Figure 16: TAG elementary trees for long distance dependencies



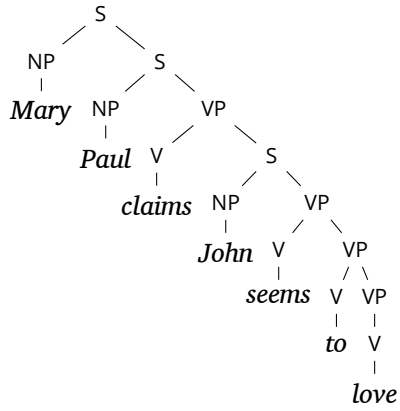


(a) Derived tree

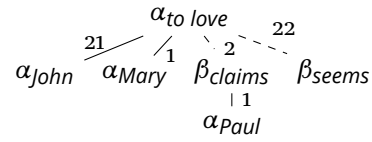


(b) Derivation tree

Figure 17: TAG analysis of *Paul claims John loves Mary*



(a) Derived tree



(b) Derivation tree

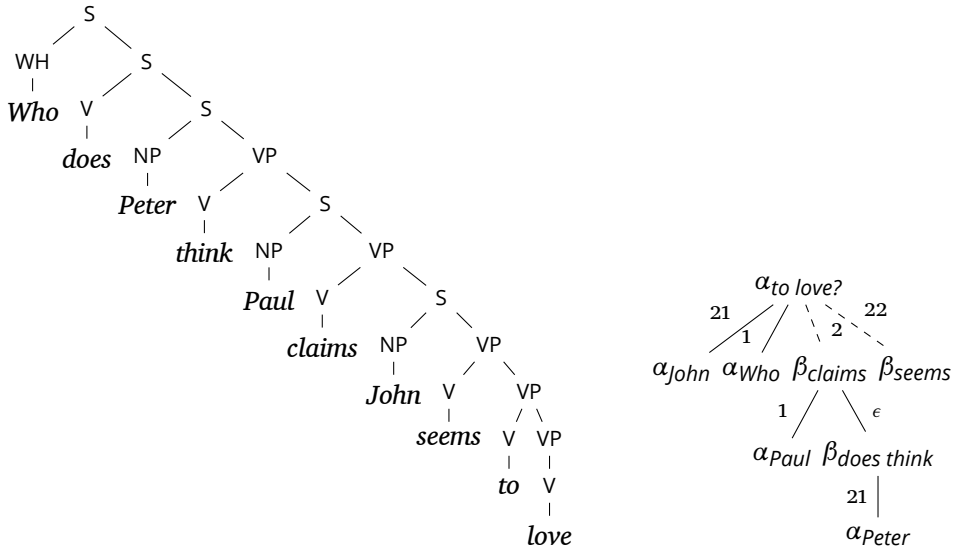
Figure 18: TAG analysis of *Mary, Paul claims John seems to love*

should scope over the **love** predicate. Moreover, the derivation tree does not specify any scoping relation between the two auxiliary trees, whereas we expect **claim** to semantically scope over **seem**.

Finally, (17) and the derivation tree of Figure 19(b) (p. 562) illustrate how an element such as a *wh*-word can scope over a whole sentence and all its predicates while providing a semantic argument to the semantically "lowest" predicate (**love**).

To semantically account for these phenomena, we first extend $\Sigma_{derivations}$ and $\mathcal{G}_{derived\ trees}$ to represent the trees of Figure 16. Table 7 (p. 563) shows the new constants and their interpretations.[23] The terms c_{17}, c_{18}, and c_{19} in (18) represent the derivation trees of

[23] Despite $\alpha_{to\ love}$ having two s nodes, its typing and its interpretation show that s adjunction is only allowed at the root node.



(a) Derived tree (b) Derivation tree

Figure 19: TAG analysis of *Who does Peter think Paul claims John seems to love*

Figure 17(b), 18(b), and 19(b) respectively. We leave it to the reader to check that the $\mathcal{G}_{\textit{derived trees}}$ interpretations of theses terms are the derived trees of the corresponding figures 17(a), 18(a), and 19(a) respectively.

(18) $c_{17} = c_{loves}\ (c_{claims}\ I_{\mathsf{S}}\ I_{\mathsf{VP}}\ c_{Paul})\ I_{\mathsf{VP}}\ c_{John}\ c_{Mary}$

$c_{18} = c_{to\ love}\ (c_{claims}\ I_{\mathsf{S}}\ I_{\mathsf{VP}}\ c_{Paul})\ (c_{seems}\ I_{\mathsf{VP}})\ c_{Mary}\ c_{John}$

$c_{19} = c_{to\ love?}\ (c_{claims}\ (c_{does\ think}\ I_{\mathsf{S}}\ I_{\mathsf{VP}}\ c_{Peter})\ I_{\mathsf{VP}}\ c_{Paul})$
$(c_{seems}\ I_{\mathsf{VP}})\ c_{who}\ c_{John}$

We now need to define the $\mathcal{G}_{sem.}$ interpretation that provides the expected semantic dependencies. Table 8 (p. 563) shows the lexical semantics fulfilling the requirements. The constant c_{loves} scopes over c_{claims} in the term $c_{17} = c_{loves}\ (c_{claims}\ I_{\mathsf{S}}\ I_{\mathsf{VP}}\ c_{Paul})\ I_{\mathsf{VP}}\ c_{John}\ c_{Mary}$, as does α_{loves} over α_{claims} in the derivation tree of Figure 17(b). However, looking at $\mathcal{G}_{sem.}(c_{loves})$ in Table 8, we observe that its first argument adv_{S} scopes over the **love** predicate. This argument actually corresponds to the meaning of the auxiliary tree adjoined at the S node of α_{loves}. When it is replaced by some actual value,

c_{who}	$: \mathsf{NP}$
	$:=_{derived\ trees}\ \gamma_{who}$
γ_{who}	$\triangleq \mathsf{NP}_1\ \mathit{who} : T$
c_{loves}	$: (\mathsf{S} \multimap \mathsf{S}) \multimap (\mathsf{VP} \multimap \mathsf{VP}) \multimap \mathsf{NP} \multimap \mathsf{NP} \multimap \mathsf{S}$
	$:=_{derived\ trees}\ \gamma_{loves}$
γ_{loves}	$\triangleq \lambda^{\mathbf{o}} S\ a\ s\ o.S\ (\mathsf{S}_2\ s\ (a\ (\mathsf{VP}_2\ (\mathsf{V}_1\ \mathit{loves})\ o)))$
	$: (T \multimap T) \multimap (T \multimap T) \multimap T \multimap T \multimap T$
$c_{to\ love}$	$: (\mathsf{S} \multimap \mathsf{S}) \multimap (\mathsf{VP} \multimap \mathsf{VP}) \multimap \mathsf{NP} \multimap \mathsf{NP} \multimap \mathsf{S}$
	$:=_{derived\ trees}\ \gamma_{to\ love}$
$\gamma_{to\ love}$	$\triangleq \lambda^{\mathbf{o}} S\ a\ o\ s.\mathsf{S}_2\ o\ S\ ((\mathsf{S}_2\ s\ (a\ (\mathsf{VP}_2\ (\mathsf{V}_1\ \mathit{to})\ (\mathsf{VP}_1\ (\mathsf{V}_1\ \mathit{love}))))))$
	$: (T \multimap T) \multimap (T \multimap T) \multimap T \multimap T \multimap T$
$c_{to\ love?}$	$: (\mathsf{S} \multimap \mathsf{S}) \multimap (\mathsf{VP} \multimap \mathsf{VP}) \multimap \mathsf{NP} \multimap \mathsf{NP} \multimap \mathsf{S}$
	$:=_{derived\ trees}\ \gamma_{to\ love?}$
$\gamma_{to\ love?}$	$\triangleq \lambda^{\mathbf{o}} S\ a\ w\ s.S\ (\mathsf{S}_2\ w\ (\mathsf{S}_2\ s\ (a\ (\mathsf{VP}_2\ (\mathsf{V}_1\ \mathit{to})\ (\mathsf{VP}_1\ (\mathsf{V}_1\ \mathit{love}))))))$
	$: (T \multimap T) \multimap (T \multimap T) \multimap T \multimap T \multimap T$
c_{claims}	$: (\mathsf{S} \multimap \mathsf{S}) \multimap (\mathsf{VP} \multimap \mathsf{VP}) \multimap \mathsf{NP} \multimap (\mathsf{S} \multimap \mathsf{S})$
	$:=_{derived\ trees}\ \gamma_{claims}$
γ_{claims}	$\triangleq \lambda^{\mathbf{o}} S\ a\ s.\lambda^{\mathbf{o}} c.S\ (\mathsf{S}_2\ s\ (a\ (\mathsf{VP}_2\ (\mathsf{V}_1\ \mathit{claims})\ c)))$
	$: (T \multimap T) \multimap (T \multimap T) \multimap T \multimap (T \multimap T)$
c_{seems}	$: (\mathsf{VP} \multimap \mathsf{VP}) \multimap (\mathsf{VP} \multimap \mathsf{VP})$
	$:=_{derived\ trees}\ \gamma_{seems}$
γ_{seems}	$\triangleq \lambda^{\mathbf{o}} a\ v.a\ (\mathsf{VP}_2\ (\mathsf{V}_1\ \mathit{seems})\ v)$
	$: (T \multimap T) \multimap (T \multimap T)$
$c_{does\ think}$	$: (\mathsf{S} \multimap \mathsf{S}) \multimap (\mathsf{VP} \multimap \mathsf{VP}) \multimap \mathsf{NP} \multimap (\mathsf{S} \multimap \mathsf{S})$
	$:=_{derived\ trees}\ \gamma_{does\ think}$
$\gamma_{does\ think}$	$\triangleq \lambda^{\mathbf{o}} S\ a\ s.\lambda^{\mathbf{o}} c.\mathsf{S}_2\ (\mathsf{V}_1\ \mathit{does})\ (S\ (\mathsf{S}_2\ s\ (a\ (\mathsf{VP}_2\ (\mathsf{V}_1\ \mathit{think})\ c))))$
	$: (T \multimap T) \multimap (T \multimap T) \multimap T \multimap (T \multimap T)$

Table 7: Interpretation by $\mathscr{G}_{derived\ trees}$

c_{who}	$:=_{sem.}\ \lambda^{\mathbf{o}} P.\mathbf{WHO}\ P$
c_{loves}	$:=_{sem.}\ \lambda^{\mathbf{o}} adv_{\mathsf{S}}\ adv_{\mathsf{VP}}\ subj\ obj.adv_{\mathsf{S}}\ (subj\ (adv_{\mathsf{VP}}\ (\lambda x.obj\ (\lambda y.\mathbf{love}\ x\ y))))$
$c_{to\ love}$	$:=_{sem.}\ \lambda^{\mathbf{o}} adv_{\mathsf{S}}\ adv_{\mathsf{VP}}\ obj\ subj.adv_{\mathsf{S}}\ (subj\ (adv_{\mathsf{VP}}\ (\lambda x.obj\ (\lambda y.\mathbf{love}\ x\ y))))$
$c_{to\ love?}$	$:=_{sem.}\ \lambda^{\mathbf{o}} adv_{\mathsf{S}}\ adv_{\mathsf{VP}}\ wh\ subj.wh\ (\lambda^{\mathbf{o}} y.adv_{\mathsf{S}}\ (subj\ (adv_{\mathsf{VP}}\ (\lambda x.\mathbf{love}\ x\ y))))$
c_{claims}	$:=_{sem.}\ \lambda^{\mathbf{o}} adv_{\mathsf{S}}\ adv_{\mathsf{VP}}\ subj\ comp.adv_{\mathsf{S}}\ (subj\ (adv_{\mathsf{VP}}\ (\lambda x.\mathbf{claim}\ x\ comp)))$
c_{seems}	$:=_{sem.}\ \lambda^{\mathbf{o}} mod\ pred.mod\ (\lambda x.\mathbf{seem}\ x\ pred)$
$c_{does\ think}$	$:=_{sem.}\ \lambda^{\mathbf{o}} adv_{\mathsf{S}}\ adv_{\mathsf{VP}}\ subj\ comp.adv_{\mathsf{S}}\ (subj\ (adv_{\mathsf{VP}}\ (\lambda x.\mathbf{think}\ x\ comp)))$

Table 8: Interpretation by $\mathscr{G}_{sem.}$ of the $\Sigma_{derivations}$ vocabulary – long distance dependencies

for instance by the interpretation of $(c_{claims}\ I_{\mathsf{S}}\ I_{\mathsf{VP}}\ c_{Paul})$, the predicate in this actual value (here **claim**) then takes scope over **love**, achieving the desired effect. The same holds for the adv_{VP} argument.

However, in $\mathscr{G}_{sem.}(c_{to\ love?})$, the *wh* argument takes scope over the whole interpretation. This argument corresponds to the meaning of the constituent to be substituted at the WH node of $\alpha_{to\ love?}$ (see Figure 16), typically $\lambda^{\mathrm{o}}P.\mathbf{WHO}\ P : (e \multimap t) \multimap t$, making **WHO** eventually take scope over all the other predicates.

Equation (19) shows that the $\mathscr{G}_{sem.}$ interpretation builds the expected semantics with the required scope inversions. In terms of lexical semantics, the analysis and the account we propose are very close to the one proposed in synchronous TAG (Nesson 2009, p. 142).

(19)

$$c_{17} = c_{loves}\ (c_{claims}\ I_{\mathsf{S}}\ I_{\mathsf{VP}}\ c_{Paul})\ I_{\mathsf{VP}}\ c_{John}\ c_{Mary}$$
$$:=_{sem.} \mathbf{claim\ paul}\ (\mathbf{love\ john\ mary})$$
$$c_{18} = c_{to\ love}\ (c_{claims}\ I_{\mathsf{S}}\ I_{\mathsf{VP}}\ c_{Paul})\ (c_{seems}\ I_{\mathsf{VP}})\ c_{Mary}\ c_{John}$$
$$:=_{sem.} \mathbf{claim\ paul}\ (\mathbf{seem\ john}\ (\lambda x.\mathbf{love}\ x\ \mathbf{mary}))$$
$$c_{19} = c_{to\ love?}\ (c_{claims}\ (c_{does\ think}\ I_{\mathsf{S}}\ I_{\mathsf{VP}}\ c_{Peter})\ I_{\mathsf{VP}}\ c_{Paul})\ (c_{seems}\ I_{\mathsf{VP}})\ c_{who}\ c_{John}$$
$$:=_{sem.} \mathbf{WHO}\ (\lambda y.\mathbf{think\ peter}\ (\mathbf{claim\ paul}\ (\mathbf{seem\ john}\ (\lambda x.\mathbf{love}\ x\ y))))$$

Remark 9. The interpretation of c_{loves}, $c_{to\ love}$, and $c_{to\ love?}$ are very close to each other. Building large scale grammars would require some factoring as can be done by lexical rules or meta-grammars (Candito 1996, 1999; Xia 2001; Xia *et al.* 2005; Crabbé *et al.* 2013). But in this article, we give the terms corresponding to the actual elementary trees that would be generated.

5.3.2 Quantification

We address in a similar way the mismatch between the scoping relation of verbal predicates over quantifiers in derivation trees and of quantifiers over verbal predicates in the logical semantic formulas. The trees of Figure 20 provide the TAG elementary trees for the TAG analysis for (20) (resp. for (21)) shown in Figure 21 (resp. in Figure 22).

(20) Everyone loves someone.

(21) Every man loves some woman.

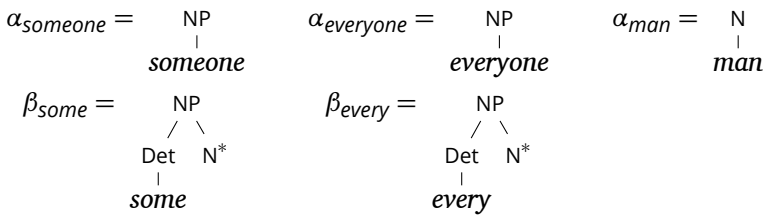
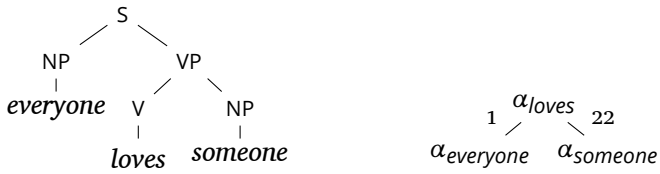


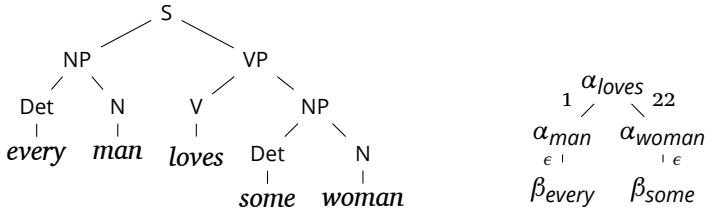
Figure 20: Determiners and quantifiers



(a) Derived tree

(b) Derivation tree

Figure 21: TAG analysis of *everyone loves someone*



(a) Derived tree

(b) Derivation tree

Figure 22: TAG analysis of *every man loves some woman*

Remark 10. We follow the standard TAG analyses for determiners (Abeillé 1990, 1993; XTAG Research Group 2001) where the latter adjoin on initial trees anchored by nouns, in order, in particular, to account for sequences of determiners (e.g., *all these ideas*) and mass nouns. While the auxiliary trees β_{some} and β_{every} of Figure 20 look unusual because the root node and the foot node do not have the same label, we can consider the label NP as a shorthand for the NP TAG category together with a positive (NP[+]) determiner feature. On the

other hand, the N label is a shorthand for the NP TAG category together with a negative (NP[−]) determiner feature. Kasper *et al.* (1995) and others (Rogers 1999; Kahane *et al.* 2000) already noted that the differences between the features on the root node and on the foot node could be reflected in allowing auxiliary trees to have different labels as root and foot nodes. While we discuss the modelling of features in TAG more generally in Section 7, the NP and N notations allow us to model the auxiliary trees of determiners with constants of the usual $\mathsf{N} \multimap \mathsf{NP}$ type (to be compared with a $\mathsf{NP}[-] \multimap \mathsf{NP}[+]$ type). While we could avoid introducing this distinction on the syntactic part of the TAG modelling, and have every node labelled with N, this distinction is semantically meaningful and records the different interpretation of N (as $e \to t$) and NP (as $(e \to t) \multimap t$) (see Table 6(a) on page 559).

The type of the constants modelling initial trees anchored by nouns has to be modified accordingly: it specifies that it requires an adjunction (an argument of type $(\mathsf{N} \multimap \mathsf{NP})$) before turning the noun into a noun phrase NP. So the type of constants (e.g., c_{man}) modelling initial trees anchored by nouns (e.g., α_{man}) is: $(\mathsf{N} \multimap \mathsf{NP}) \multimap \mathsf{NP}$. It corresponds to only keeping the constants that can indeed be used in actual derivations. For each noun, we could instead introduce two constants with the following types: $(\mathsf{NP}[-] \multimap \mathsf{NP}[-]) \multimap \mathsf{NP}[-] = (\mathsf{N} \multimap \mathsf{N}) \multimap \mathsf{N}$ and $(\mathsf{NP}[-] \multimap \mathsf{NP}[+]) \multimap \mathsf{NP}[+] = (\mathsf{N} \multimap \mathsf{NP}) \multimap \mathsf{NP}$, but since there is no other constant that uses $\mathsf{NP}[-] = \mathsf{N}$ as a type for its arguments (i.e., substitution nodes),[24] we only keep the constant with the last type.

The derivation trees of Figure 21 and 22 are again such that the elementary tree of the verb predicate dominates the other elementary trees, while the respective scopes of their semantic contributions are in the reverse order. To show how this apparent mismatch can be dealt with, we extend $\Sigma_{derivations}$ with the constants of Table 9. This table also provides the interpretation of these constants by $\mathscr{G}_{derived\ trees}$, modelling the elementary trees of Figure 20. The terms of (22) belong to $\mathscr{A}(\mathscr{G}_{derived\ trees})$ and represent the derivation trees of Figure 21

[24] This of course depends on the grammar. In any case, if there were such a constant, it would not allow for performing first the adjunction of a determiner on the noun.

Constants of $\Sigma_{derivations}$		Interpretation by $\mathscr{G}_{derived\ trees}$
c_{man}	$: (\mathsf{N} \multimap \mathsf{NP}) \multimap \mathsf{NP}$	$\lambda^{o} d.d\ (\mathsf{N}_1 \textit{man})$
$c_{someone}$	$: \mathsf{NP}$	$\mathsf{NP}_1\ \textit{someone}$
$c_{everyone}$	$: \mathsf{NP}$	$\mathsf{NP}_1\ \textit{everyone}$
c_{some}	$: \mathsf{N} \multimap \mathsf{NP}$	$\lambda^{o} n.\mathsf{NP}_2\ (\mathsf{Det}_1\ \textit{some})\ n$
c_{every}	$: \mathsf{N} \multimap \mathsf{NP}$	$\lambda^{o} n.\mathsf{NP}_2\ (\mathsf{Det}_1\ \textit{every})\ n$

Table 9:
Interpretation of $\Sigma_{derivations}$ constants by $\mathscr{G}_{derived\ trees}$

and 22. Equation (23) shows they are interpreted as the derived trees of the same figures.

(22)
$$c_{21} = c_{loves}\ I_{\mathsf{S}}\ I_{\mathsf{VP}}\ c_{everyone}\ c_{someone}$$
$$c_{22} = c_{loves}\ I_{\mathsf{S}}\ I_{\mathsf{VP}}\ (c_{man}\ c_{every})\ (c_{woman}\ c_{some})$$

(23)
$$c_{21} :=_{derived\ trees} \mathsf{S}_2\ (\mathsf{NP}_1\ \textit{everyone})\ (\mathsf{VP}_2\ (\mathsf{V}_1\ \textit{loves})\ (\mathsf{NP}_1\ \textit{someone}))$$
$$c_{22} :=_{derived\ trees} \mathsf{S}_2\ (\mathsf{NP}_2\ (\mathsf{Det}_1\ \textit{every})\ (\mathsf{N}_1\ \textit{man}))\ (\mathsf{VP}_2\ (\mathsf{V}_1\ \textit{loves})\ (\mathsf{NP}_2\ (\mathsf{Det}_1\ \textit{some})\ (\mathsf{N}_1\ \textit{woman})))$$

Then we extend $\mathscr{G}_{sem.}$ with the interpretations of these new constants of $\Sigma_{derivations}$ as terms of $\Lambda(\Sigma_{logic})$ (Table 10). The semantic interpretations of the terms c_{21} and c_{22} are then as expected, as (24) shows.

(24)
$$c_{21} :=_{sem.} \forall x.(\mathbf{human}\ x) \Rightarrow (\exists y.(\mathbf{human}\ y) \wedge (\mathbf{love}\ x\ y))$$
$$c_{22} :=_{sem.} \forall x.(\mathbf{man}\ x) \Rightarrow (\exists y.(\mathbf{woman}\ y) \wedge (\mathbf{love}\ x\ y))$$

c_{man}	$:=_{sem.} \lambda^{o} Q.\lambda^{o} q.Q\ \mathbf{man}\ q$
$c_{someone}$	$:=_{sem.} \lambda^{o} Q.\exists x.(\mathbf{human}\ x) \wedge (Q\ x)$
$c_{everyone}$	$:=_{sem.} \lambda^{o} Q.\forall x.(\mathbf{human}\ x) \Rightarrow (Q\ x)$
c_{some}	$:=_{sem.} \lambda^{o} P\ Q.\exists x.(P\ x) \wedge (Q\ x)$
c_{every}	$:=_{sem.} \lambda^{o} P\ Q.\forall x.(P\ x) \Rightarrow (Q\ x)$

Table 10:
Interpretation by $\mathscr{G}_{sem.}$ of the $\Sigma_{derivations}$ vocabulary – quantification

This shows how to use the derivation tree as a pivot towards the semantic representation of an expression. The (lexical) semantic interpretation of the terms labelling the nodes of the derivation tree encodes, when necessary, the inversion of the scope of the elements. This is reminiscent of the transformation of derivation trees

into semantic dependency graphs of Candito and Kahane (1998) or Kallmeyer and Kuhlmann (2012). To this end, the latter implements a tree transduction-based approach (macro-tree transduction). Maskharashvili and Pogodalla (2013) discuss the relation with the present approach, relying on the encoding of macro-tree transduction within second-order ACGs (Yoshinaka 2006).

Remark 11. There are several ways to get the object scope reading. So far, the relative scopes of the subject and the object are bound to the semantic interpretation of the verb (see the semantic interpretation of c_{loves} in Table 8 on page 563). So a possibility consists in introducing a new constant c^{ows}_{loves} whose semantic interpretation reverses the scope, as Equation (25) shows.

(25) $c^{\text{ows}}_{loves} :=_{sem}$
$\lambda^{o} adv_{S}\ adv_{VP}\ subj\ obj.adv_{S}\ (obj\ (\lambda y.subj\ (adv_{VP}\ (\lambda x.\mathbf{love}\ x\ y))))$

Another possibility, that would go beyond what is introduced in this article, would be to use Multi-Component TAG (MCTAG: Weir 1988) as Williford (1993) proposes. In both cases, some care should be taken in order not to introduce spurious ambiguities. In Section 8.3, we provide another modelling that allows us to get this reading, and we relate it to other approaches.

5.3.3 Multiple adjunctions

The representation of TAG derivation trees as abstract terms of an ACG corresponds to the standard notion of derivation trees (Vijay-Shanker 1987). Schabes and Shieber (1994) call it *dependent* and advocate for an alternative *independent* notion. With dependent derivations, and in our approach, multiple adjunction on the same node is forbidden. So the analysis of (26), using the trees of Figure 23, requires first the adjunction of β_{big} into β_{black}, and the adjunction of the result into α_{dog}. Figure 25(a) shows the resulting derivation tree. On the other hand, the independent adjunction shown in Figure 25(b) only specifies that both adjectives adjoin at the N node of the initial tree, corresponding to both the derived tree of Figure 24(a) and the derived tree of Figure 24(b).

(26) big black dog

(27) black big dog

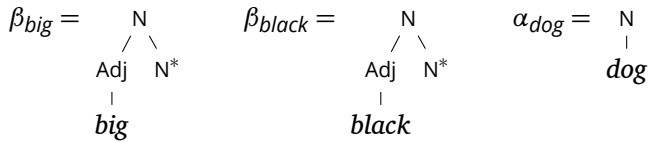
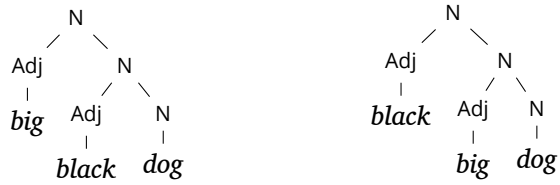


Figure 23: TAG elementary trees for adjectives



(a) Derived tree for (26)

(b) Derived tree for (27)

Figure 24: Alternative notions of derived trees

(a) Dependent derivation tree

(b) Independent derivation tree

Figure 25: Alternative notions of derivation trees

Schabes and Shieber (1994) present several arguments in favour of multiple adjunction for auxiliary trees encoding modification (as opposed to auxiliary trees encoding predication) and independent derivations. We only discuss here the semantic argument they provide.[25] The main concern again has to do with the relation between derivation trees and semantic dependencies. The dependent derivation of Figure 25(a) reflects "cascaded modifications" of the head, rather

[25] The two other main arguments relate to the addition of adjoining constraints and to the addition of statistical parameters. Adding the latter to ACGs as a general framework is ongoing work, and the effects on the particular case of the TAG into ACG encoding will be considered from this perspective (Huot 2017). The argument about adjoining constraints that fail to escape intervening adjunctions is not related to the syntax-semantics interface and deserves a discussion that is beyond the scope of this article. For instance, the example of the [+] determiner feature (Section 5.3.2) that can percolate from the determiner (outmost adjunction) to the noun, despite the intervening adjunctions of the adjectives, shows that selectional restrictions can be implemented with long-distance effects.

than more expected "separate modifications", the latter being only available through multiple adjunction. We show that we can actually achieve this effect in our framework, without multiple adjunction, by specifying a semantic interpretation for adjectives that encodes such a behavior.

We consider the extension of $\Sigma_{derivations}$ with the constants of Table 11 and their interpretations by $\mathscr{G}_{derived\ trees}$ and $\mathscr{G}_{sem.}$ of Table 12. The types of the the constants modelling adjectives follow the types proposed for constants modelling nouns. The modification they introduce builds a NP from a N, and can itself take a (N ⊸ NP) modification (adjunction) into account. Consequently, they are of type (N ⊸ NP) ⊸ N ⊸ NP. As we did for the types of the constants modelling nouns, we could enumerate the possible types taking the determiner feature into account. Because the adjunction of an adjective does not change the determiner feature, its value at the root node of the auxiliary tree only depends on what is possibly adjoined to it. So we could have four constants with the following types: (NP[−] ⊸ NP[−]) ⊸ (NP[−] ⊸ NP[−]), (NP[−] ⊸ NP[+]) ⊸ (NP[−] ⊸ NP[+]), (NP[+] ⊸ NP[−]) ⊸ (NP[+] ⊸ NP[−]), and (NP[+] ⊸ NP[+]) ⊸ (NP[+] ⊸ NP[+]). But if we do not provide a term for a fake adjunction (NP[−] ⊸ NP[−]), (NP[+] ⊸ NP[+]), or (NP[+] ⊸ NP[−]) (as in the example grammar we have), such terms can never be used in a S derivation. So we only keep the constants that have type (NP[−] ⊸ NP[+]) ⊸ (NP[−] ⊸ NP[+]) = (N ⊸ NP) ⊸ N ⊸ NP.

Equation (28) shows the interpretations of the term $c_{25} : (\mathsf{N} \multimap \mathsf{NP}) \multimap \mathsf{NP}$ (an expression missing a determiner of type (N ⊸ NP) to provide a NP) that encodes the derivation tree of Figure 25(a) (p. 569).

Table 11: $\Sigma_{derivations}$ additional constants

c_{big}	$: (\mathsf{N} \multimap \mathsf{NP}) \multimap \mathsf{N} \multimap \mathsf{NP}$
c_{black}	$: (\mathsf{N} \multimap \mathsf{NP}) \multimap \mathsf{N} \multimap \mathsf{NP}$
c_{dog}	$: (\mathsf{N} \multimap \mathsf{NP}) \multimap \mathsf{NP}$

Table 12: Interpretation by $\mathscr{G}_{sem.}$ and $\mathscr{G}_{derived\ trees}$ of the $\Sigma_{derivations}$ vocabulary – multiple adjunction

c_{big}	$:=_{derived\ trees} \lambda^{\circ} a\ n.a\ (\mathsf{N}_2\ (\mathsf{Adj}_1\ \mathit{big})\ n)$
c_{black}	$:=_{derived\ trees} \lambda^{\circ} a\ n.a\ (\mathsf{N}_2\ (\mathsf{Adj}_1\ \mathit{black})\ n)$
c_{dog}	$:=_{derived\ trees} \lambda^{\circ} d.d\ (\mathsf{N}_1\ \mathit{dog})$
c_{big}	$:=_{sem.} \lambda^{\circ} Q\ n.\lambda^{\circ} q.Q\ (\lambda x.(n\ x) \wedge (\mathbf{big}\ x))\ q$
c_{black}	$:=_{sem.} \lambda^{\circ} Q\ n.\lambda^{\circ} q.Q\ (\lambda x.(n\ x) \wedge (\mathbf{black}\ x))\ q$
c_{dog}	$:=_{sem.} \lambda^{\circ} Q.\lambda^{\circ} q.Q\ \mathbf{dog}\ q$

The interpretation by $\mathscr{G}_{sem.}$ indeed provides a separate modification of the same variable x as argument both of **big** and **black** (a similar account would also be available in synchronous TAG).

$$
\begin{aligned}
& c_{25} = \lambda^{o} D.c_{dog}\ (c_{black}\ (c_{big}\ D)) \\
(28)\qquad & :=_{derived\ trees}\ \lambda^{o} D.D\ (\mathsf{N}_2\ (\mathsf{Adj}_1\ \textit{big})\ (\mathsf{N}_2\ (\mathsf{Adj}_1\ \textit{black})\ (\mathsf{N}_1\ \textit{dog}))) \\
& :=_{sem.}\ \lambda^{o} D.\lambda^{o} q.D\ (\lambda x.((\mathbf{big}\ x) \wedge (\mathbf{black}\ x)) \wedge (\mathbf{dog}\ x))\ q
\end{aligned}
$$

Remark 12. By not introducing a constant $I_{\mathsf{N}} : \mathsf{N} \multimap \mathsf{NP}$ in $\Sigma_{derivations}$, we require actual adjunctions of determiners (of type $(\mathsf{N} \multimap \mathsf{NP})$, e.g., c_{some}) on nouns or on nouns modified by adjectives.

6 COMPLETING THE TAG INTO ACG ENCODING

So far, the abstract signatures we used, in particular $\Sigma_{derivations}$, introduce constants that are of order strictly greater than 2. This comes in particular from the modelling of auxiliary trees as functions (typically of type $X \multimap X$), hence from having constants of higher-order type modelling the ability of a tree to take an auxiliary tree as an argument. From a theoretical point of view, we know this encoding cannot faithfully model TAG: TAG languages are polynomially parsable, and 3rd-order ACGs can generate languages in NP. Remark 8 (p. 557) gives an example of an unexpected result of this encoding: there is no way to distinguish the $\mathsf{S} \multimap \mathsf{S}$ type of an abstract constant modelling an auxiliary tree of foot node S from an abstract constant modelling an elementary tree of root S with a substitution node S.

6.1 *A vocabulary for TAG derivations: the Σ_{TAG} signature*

In order to allow for the distinction between these types, we introduce *atomic types* (e.g., S_A) that will be interpreted as the modifier types of the constants modelling auxiliary trees. So in addition to the ACG $\mathscr{G}_{derived\ trees} = \langle \Sigma_{derivations}, \Sigma_{trees}, \mathscr{L}_{derived\ trees}, \mathsf{S} \rangle$, we also define a higher-order signature $\Sigma_{TAG} = \langle A_{TAG}, C_{TAG}, \tau_{TAG} \rangle$ such that $A_{TAG} = A_{derivations} \cup \bigcup_{X \in A_{derivations}} X_A$.

For any $\delta_{anchor} \in \mathcal{I} \cup \mathcal{A}$ an elementary tree of $\mathcal{G}$, C_{anchor} is a constant in C_{TAG} with type $X_A^1 \multimap \cdots \multimap X_A^n \multimap Y^1 \multimap \cdots \multimap Y^m \multimap \alpha$ where:

- the X^i are the labels of the n internal nodes of δ_{anchor} labelled with a non-terminal where an adjunction is possible (by convention we use the breadth-first traversal);

- the Y^i are the labels of the m leaves of δ_{anchor} labelled with non-terminals, *not counting the foot node if δ_{anchor} is an auxiliary tree*, of δ_{anchor} (by convention, we use the left-right order traversal);
- let Z be the label of the root node of δ_{anchor}. $\alpha = Z$ if $\delta_{anchor} \in \mathcal{I}$ is an initial tree, and $\alpha = Z_A$ with Z the label of the root node if $\delta_{anchor} \in \mathcal{A}$ is an auxiliary tree.

From the elementary trees of Figure 2 (p. 533), for instance, we get the constants typed as (29) shows.

$$\begin{aligned} C_{sleeps} &: \mathsf{S}_A \multimap \mathsf{VP}_A \multimap \mathsf{NP} \multimap \mathsf{S} \\ C_{John} &: \mathsf{NP} \\ C_{seemingly} &: \mathsf{VP}_A \multimap \mathsf{VP}_A \end{aligned} \tag{29}$$

Moreover, for each non-terminal X of the TAG grammar where an adjunction can occur, we also define $I_X : X_A$. These constants play a similar role as the I_X constants in $\Sigma_{derivations}$: they are used when a TAG derivation does not involve adjunctions on sites where it would be possible to have them.

Then the set of typed constants of Σ_{TAG} is $C_{TAG} = \{C_{anchor} | \ \delta_{anchor} \in \mathcal{I} \cup \mathcal{A}\} \cup \{I_X | \ X \in V^0\}$ and τ_{TAG} is the associated typing function defined as above. The typing provided by Σ_{TAG} now distinguishes the type of the encoding of an elementary tree of root S with a substitution node S (type $\mathsf{S} \multimap \mathsf{S}$) and the encoding of an auxiliary tree with root and foot nodes labelled by S (type S_A, see Remark 8, p. 557).

We now need to relate the terms of $\Lambda(\Sigma_{TAG})$ to the terms of $\Lambda(\Sigma_{derivations})$ by a suitable interpretation.

6.2 *Interpreting Σ_{TAG} into $\Lambda(\Sigma_{derivations})$: the $\mathcal{G}_{TAG}$ ACG*

We now can relate Σ_{TAG} and $\Lambda(\Sigma_{derivations})$ through a new ACG $\mathcal{G}_{TAG} = \langle \Sigma_{TAG}, \Sigma_{derivations}, \mathcal{L}_{TAG}, \mathsf{S} \rangle$ where $\mathcal{L}_{TAG}$ is such that:

- for all $\alpha \in A_{TAG}$, if $\alpha = X_A$ then $\mathcal{L}_{TAG}(\alpha) = \mathcal{L}_{TAG}(X_A) = X'' \multimap X'$ with, most of the time, $X = X' = X''$ (see footnote 18 (p. 553) and Remark 13, next page), otherwise $\alpha = X \in A_{derivations}$ and $\mathcal{L}_{TAG}(\alpha) = \mathcal{L}_{TAG}(X) = X$;
- for all $C_{anchor} \in C_{TAG}$, $\mathcal{L}_{TAG}(C_{anchor}) = c_{anchor}$.

By construction of the constants $c_{anchor} \in C_{derivations}$ (Section. 4.3), and by construction of the constants $C_{anchor} \in C_{TAG}$, $\mathscr{L}_{TAG}$ is well defined.

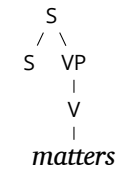
Table 13 sums up the constants corresponding to the elementary trees introduced so far as well as their interpretations. Because constants are interpreted as constants, the terms of $\Lambda(\Sigma_{TAG})$ and their interpretations are isomorphic. However, some terms of $\Lambda(\Sigma_{derivations})$ have no antecedent in $\mathscr{L}_{TAG}$. For instance, the term

Types and constants of Σ_{TAG}		Their interpretations in $\Lambda(\Sigma_{derivations})$	
NP		NP	
S		S	
VP		VP	
N		N	
WH		WH	
VP_A		VP ⊸ VP	
S_A		S ⊸ S	
N_A		N ⊸ NP	
C_{John}	: NP	c_{John}	: NP
C_{sleeps}	: $S_A \multimap VP_A \multimap NP \multimap S$	c_{sleeps}	: $(S \multimap S) \multimap (VP \multimap VP) \multimap NP \multimap S$
$C_{seemingly}$	: $VP_A \multimap VP_A$	$c_{seemingly}$	: $(VP \multimap VP) \multimap VP \multimap VP$
$C_{usually}$	: $VP_A \multimap VP_A$	$c_{usually}$	: $(VP \multimap VP) \multimap VP \multimap VP$
C_{hence}	: $S_A \multimap S_A$	c_{hence}	: $(S \multimap S) \multimap S \multimap S$
I_S	: S_A	I_S	: S ⊸ S
I_{VP}	: VP_A	I_{VP}	: VP ⊸ VP
$C_{matters}$	: $VP_A \multimap S \multimap S$	$c_{matters}$	: $(VP \multimap VP) \multimap S \multimap S$
C_{who}	: NP	c_{who}	: NP
C_{loves}	: $S_A \multimap VP_A \multimap NP \multimap NP$	c_{loves}	: $(S \multimap S) \multimap (VP \multimap VP) \multimap NP \multimap NP \multimap S$
$C_{to\ love}$	: $S_A \multimap VP_A \multimap NP \multimap NP$	$c_{to\ love}$	: $(S \multimap S) \multimap (VP \multimap VP) \multimap NP \multimap NP \multimap S$
$C_{to\ love?}$	: $S_A \multimap VP_A \multimap NP \multimap NP$	$c_{to\ love?}$	: $(S \multimap S) \multimap (VP \multimap VP) \multimap NP \multimap NP \multimap S$
C_{claims}	: $S_A \multimap VP_A \multimap NP \multimap S_A$	c_{claims}	: $(S \multimap S) \multimap (VP \multimap VP) \multimap NP \multimap (S \multimap S)$
C_{seems}	: $VP_A \multimap VP_A$	c_{seems}	: $(VP \multimap VP) \multimap (VP \multimap VP)$
$C_{does\ think}$	: $S_A \multimap VP_A \multimap NP \multimap S_A$	$c_{does\ think}$	: $(S \multimap S) \multimap (VP \multimap VP) \multimap NP \multimap (S \multimap S)$
C_{man}	: $N_A \multimap NP$	c_{man}	: $(N \multimap NP) \multimap NP$
$C_{someone}$	: NP	$c_{someone}$	: NP
$C_{everyone}$	: NP	$c_{everyone}$	: NP
C_{some}	: N_A	c_{some}	: N ⊸ NP
C_{every}	: N_A	c_{every}	: N ⊸ NP
C_{big}	: $N_A \multimap N_A$	c_{big}	: $(N \multimap NP) \multimap N \multimap NP$
C_{black}	: $N_A \multimap N_A$	c_{black}	: $(N \multimap NP) \multimap N \multimap NP$
C_{dog}	: $N_A \multimap NP$	c_{dog}	: $(N \multimap NP) \multimap NP$

Table 13: Σ_{TAG} constants and their interpretation by $\mathscr{L}_{TAG}$

$c_{sleeps}\ (c_{matters}\ I_{\mathsf{VP}})\ I_{\mathsf{VP}}\ c_{John} : \mathsf{S} \in \mathscr{A}(\mathscr{G}_{derived\ trees})$, where $c_{matters} : (\mathsf{VP} \multimap \mathsf{VP}) \multimap \mathsf{S} \multimap \mathsf{S}$ corresponds to the initial tree $\alpha_{matters}$ of Figure 26, as in *To arrive on time matters considerably* (see XTAG Research Group 2001, Section 6.31), has no antecedent. This is because the type of $c_{matters}\ I_{\mathsf{VP}} : \mathsf{S} \multimap \mathsf{S}$ in $\mathscr{G}_{derived\ trees}$, while it encodes an initial tree, is the same as the type of a term encoding an adjunction on a S node (see Remark 8 p. 557). But this is not true anymore at the level of $\mathscr{G}_{TAG}$ where $C_{matters}\ I_{\mathsf{VP}} : \mathsf{S} \multimap \mathsf{S}$ but the type of a term encoding an adjunction on a S node is now S_A.

Figure 26:
Initial tree for *matters*
(the S leaf is a substitution node)



So Σ_{TAG} allows us to add control on the admissible derivation structures that $\Sigma_{derivations}$ can provide. The general architecture is now the one of Figure 15 (p. 557). Moreover, this architecture allows us to provide, by function composition, a semantic interpretation to the constants of Σ_{TAG}. Interestingly, this semantic interpretation derives from the more general constructions that $\Sigma_{derivations}$ enables.

Σ_{TAG} strictly follows the abstract signature definition proposed by de Groote (2002) to encode the syntactic part of TAGs. The correctness of our encoding follows from the fact that the ACG $\mathscr{G}_{derived\ trees} \circ \mathscr{G}_{TAG}$ we get by function composition is the ACG defined by de Groote (2002).

Remark 13. Because the modelling of adjunction is now controlled by the interpretation of the types X_A from Σ_{TAG}, we see that we can have more freedom in the type that is given in $\Sigma_{derivations}$. For instance, we can set $\mathsf{N}_A :=_{TAG} \mathsf{N} \multimap \mathsf{NP}$. We can use even more complex interpretations if it helps explaining the semantic interpretation. For instance, in Section 7.2 we introduce a type $\mathsf{S}'_A :=_{TAG} (\mathsf{NP} \multimap \mathsf{S}) \multimap \mathsf{S}$ to model control verbs.

We can now consider the ACG $\mathscr{G}_{yield} \circ \mathscr{G}_{derived\ trees} \circ \mathscr{G}_{TAG}$, that interprets terms of $\Lambda(\Sigma_{TAG})$ into strings, and the ACG $\mathscr{G}_{sem.} \circ \mathscr{G}_{TAG}$, that interprets terms of $\Lambda(\Sigma_{TAG})$ into logical formulas (see Figure 15,

p. 557). Because Σ_{TAG} is second-order, these two ACGs are second-order (while $\mathcal{G}_{yield} \circ \mathcal{G}_{derived\ trees}$ and $\mathcal{G}_{sem.}$ are not, since $\Sigma_{derivations}$ is not second-order). Hence the parsing result applies and we may parse terms with ACGs that have Σ_{TAG} as abstract vocabulary, in particular with the ACG toolkit. The ACG example files we provide can, for instance, parse the string *every* + *big* + *black* + *dog* + *usually* + *barks*. It can also parse the logical formula $\forall x.(((\mathbf{dog}\ x) \wedge (\mathbf{black}\ x)) \wedge (\mathbf{big}\ x)) \Rightarrow (\mathbf{usually}\ (\mathbf{bark}\ x))$. Note that, as a λ-term, a logical formula can generally not be replaced by a logically equivalent formula. This is an instance of the problem of logical-form equivalence (Shieber 1993) that will need to be addressed, for instance using sets of λ-terms as input (Kanazawa 2017, Section 4.2). More examples are available in the example files.

7 ADJOINING CONSTRAINTS AND FEATURES

It is part of the TAG formalism to specify if an internal node may, may not, or must receive any adjunction. The latter case is called an *obligatory adjoining (OA)* constraint. In case an internal node can be subject to an adjunction operation, it is also possible to specify a restricted set of auxiliary trees, with relevant root and foot nodes, that can adjoin. This constraint is called a *selective adjoining (SA)* constraint. There are different ways to specify such constraints in TAG. One is to add *features* to the formalism. ACGs do not provide a concise way to express the abstract representation of type constraints that features offer. There have been some proposals with dependent types (de Groote and Maarek 2007; de Groote *et al.* 2007; Pompigne 2013), but the underlying calculus does not have the expected good properties. So selection restriction has to be expressed by introducing as many types as necessary (see Section 5.3.2 for determiners and Section 5.3.3 for adjectives). To avoid the drawback of a growing size of the grammar, the addition of features, in particular morpho-syntactic ones, to ACGs remains desirable.

Note, however, that we do not want to consider features that are used to model the syntax-semantics interface (Kallmeyer and Romero 2004, 2008), since we use the interpretation of derivation trees instead. We discuss the relation between the two approaches in Section 7.3.

7.1 *Obligatory adjoining constraints*

Section 5.3.3 presents an instance of an adjoining constraint, namely an obligatory adjoining constraint. In order to form a NP, a determiner of type $(\mathsf{N} \multimap \mathsf{NP})$ needs to be adjoined (directly or through adjectival modifications) into a noun. The obligatory nature of the adjunction is reflected by the fact that the abstract vocabulary does not provide any constant $I_{\mathsf{N}} : \mathsf{N} \multimap \mathsf{NP}$ simulating a fake adjunction.

7.2 *Selective adjoining constraints*

In Section 5.3.2, we saw an instance of using features in a TAG analysis: noun phrases can receive a determiner feature $[+]$ or $[-]$ indicating whether they are determined. The ACG way to account for this distinction instead consists in introducing different (atomic) types. This corresponds to specifying local adjunction constraints by *enumeration* as in TAG and contrary to Feature-based TAG (FTAG: Vijay-Shanker and Joshi 1988, 1991).

As noticed in Remark 13 (p. 574), the key here is to model auxiliary trees using an atomic type X_A, so that the ACG is second-order, and to interpret this type as a functional type $X'' \multimap X'$ of $\Sigma_{\textit{derivations}}$, without the actual requirement that X'' and X' are atomic or that $X'' = X' = X$. We illustrate such an encoding with the TAG analysis of control verbs.

TAG analyzes a sentence such as (30) with an adjunction of the subject control verb *wants* on the reduced clause *to sleep* as Figure 28 shows. Figure 27 presents the elementary trees of the control verb and of the infinitive clause. This is similar to representing infinitive clauses as clauses without subjects (Abeillé 2002). For the sake of simplicity, we directly represent such a clause as an elementary tree with a PRO node.

Control and adjunction are enforced using a control feature on the S root node of the complement tree (`control` in XTAG (XTAG Research Group 2001, p. 98), or some semantic index `idx` (Gardent and Parmentier 2005; Gardent 2008)) that is to be provided by the foot node of the auxiliary tree (the control verb) which is adjoined. Moreover, in the auxiliary tree, the control feature on the foot node is co-indexed with a control feature on the subject NP (for subject control, as in (30)) or on the object NP (for object control). Figure 28 shows

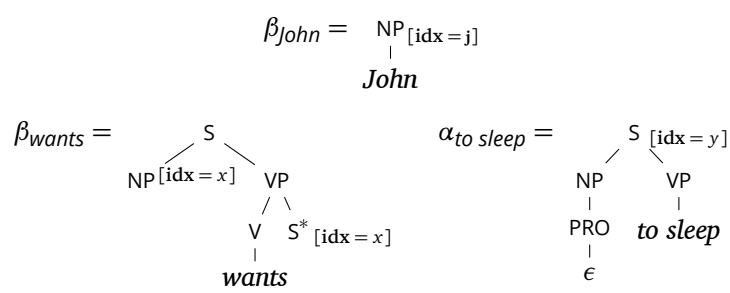
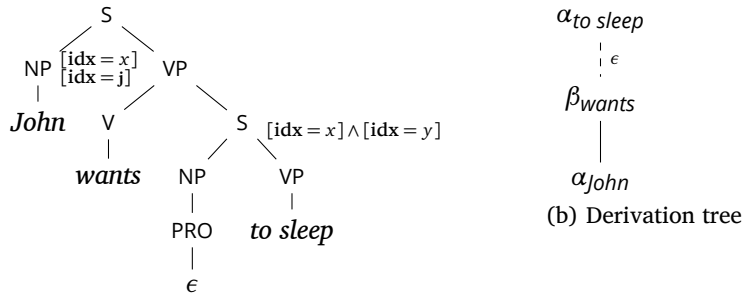


Figure 27: Elementary trees for control verbs



(a) Derived tree

(b) Derivation tree

Figure 28: TAG analysis of control

the derived tree, where, by unification of the top and bottom features, we eventually get $x = y = \mathbf{j}$, and the derivation tree for (30).

(30) John wants to sleep

In our ACG encoding, we model S nodes with a control feature with the functional type NP ⊸ S, expressing that such a clause is missing its subject. Consequently, the type of the constant c_{wants} that models the auxiliary tree β_{wants} is (S ⊸ S) ⊸ (VP ⊸ VP) ⊸ NP ⊸ (NP ⊸ S) ⊸ S. The end part (NP ⊸ S) ⊸ S of this type corresponds to the functional interpretation of the adjunction of control verbs, modelled at the Σ_{TAG} level with the atomic type S'_A. The difference between the type (NP ⊸ S) of the argument and the type S of the result corresponds to the different feature set attached to the root node and to the foot node of the auxiliary tree.

Then we model the feature sharing between the subject NP and the S foot node of the control verb in the semantic interpretation of the latter, as the second line of Table 16 shows: the first argument x of **want** x (*pred* $(\lambda^{o} P.P\ x)$) also appears (type raised as $(\lambda^{o} P.P\ x)$) as

Table 14: $\mathscr{G}_{TAG}$ extension – control verbs

Types and constants of Σ_{TAG}		Their interpretations in $\Lambda(\Sigma_{derivations})$	
S'_A		$(NP \multimap S) \multimap S$	
C_{wants}	$: S_A \multimap VP_A \multimap NP \multimap S'_A$	c_{wants}	$: (S \multimap S) \multimap (VP \multimap VP) \multimap NP \multimap (NP \multimap S) \multimap S$
$C_{to\ sleep}$	$: S'_A \multimap S$	$c_{to\ sleep}$	$: ((NP \multimap S) \multimap S) \multimap S$

Table 15: $\mathscr{G}_{derived\ trees}$ extension – control verbs

$$c_{wants} :=_{derived\ trees} \lambda^{o} adv_S\ adv_{VP}\ subj\ pred.\ adv_S\ (S_2\ subj\ (adv_{VP}\ (VP_2\ (V_1\ wants)\ (pred\ (PRO_1\ \epsilon)))))$$
$$c_{to\ sleep} :=_{derived\ trees} \lambda^{o} cont.cont(\lambda^{o} subj.S_2\ (NP_1\ subj)\ (VP_2\ (V_1\ to)\ (VP_1\ sleep)))$$

Table 16: Σ_{logic} and $\mathscr{G}_{sem.}$ extension – control verbs

$$\mathbf{want} : e \multimap t \multimap t$$
$$c_{wants} :=_{sem.} \lambda^{o} adv_S\ adv_{VP}\ subj\ pred.adv_S\ (subj\ (adv_{VP}\ (\lambda x.\mathbf{want}\ x\ (pred\ (\lambda^{o} P.P\ x)))))$$
$$c_{to\ sleep} :=_{sem.} \lambda^{o} cont.cont(\lambda^{o} subj.subj\ (\lambda x.\mathbf{sleep}\ x))$$

the argument of *pred*, the latter corresponding to the semantics of the infinitive clause without subject.

Let C_{28} of $\Lambda(\Sigma_{TAG})$ in (31) represent the derivation tree of Figure 28, and let c_{28} be its interpretation in $\Lambda(\Sigma_{derivations})$. We can further interpret c_{28} in $\Lambda(\Sigma_{trees})$ (resp. in $\Lambda(\Sigma_{logic})$) in order to have a term representing the associated derived tree (resp. semantics).

$$\begin{aligned} C_{28} &= C_{to\ sleep}\ (C_{wants}\ I_S\ I_{VP}\ C_{John}) \\ &:=_{derivations} c_{28} \\ c_{28} &= c_{to\ sleep}\ (c_{wants}\ I_S\ I_{VP}\ c_{John}) \\ &:=_{derived\ trees} S_2\ (NP_1\ John)\ (VP_2\ (V_1\ wants)\ (S_2\ (NP_1\ (PRO_1\ \epsilon))\ (VP_2\ (V_1\ to)\ (VP_1\ sleep))))) \\ &:=_{sem.} \mathbf{want\ john}\ (\mathbf{sleep\ john}) \end{aligned} \tag{31}$$

7.3 *Feature sharing and semantic computation*

As the previous section shows, features in TAG are taken into account in the ACG encoding using the typing discipline on the one hand, and using the (semantic) interpretation on the other hand, in particular when some value has to be shared in order to express the modifications performed by adjunction operations.

Unification based approaches to semantic construction in TAG typically rely on feature sharing (Gardent and Kallmeyer 2003; Gardent and Parmentier 2005; Kallmeyer and Romero 2008) in order to

compositionally build the semantic representation of a sentence. In our approach, the semantic representation results from the interpretation of the derivation tree.

However, Vijay-Shanker and Joshi (1988, p. 718) already noticed that "[t]his treatment [of variable instantiation on adjunction] can be obtained if we think of the auxiliary tree as corresponding to functions over feature structures (by λ-abstracting the variable corresponding to the feature structure for the tree that will appear below the foot node). Adjunction corresponds to applying this function to the feature structure corresponding to the subtree below the node where [it] takes place". This is precisely the view we adopt here. While the typing exerts control over the admissible derivation structures, the associated computations are managed using interpretations, to compute the derived trees as well as the logical formulas.

8 DERIVATION TREES AND SEMANTIC INTERPRETATIONS

Looking at Figure 15 (p. 557), we can consider each of the sets of λ-terms as independent combinatorial systems of the grammar architecture that Jackendoff (2002) describes: "Language comprises a number of independent combinatorial systems which are aligned with each other by means of a collection of interface systems. Syntax is among the combinatorial systems, but far from the only one".

Among those systems, $\Lambda(\Sigma_{derivations})$ and $\Lambda(\Sigma_{TAG})$ play a central role as their structures are the ones that are interpreted as derived trees (and as strings, by functional composition) and as logical formulas. This is *not* the role of the syntactic trees of $\Lambda(\Sigma_{trees})$. This emphasises that the relevant syntactic algebra to provide compositional analyses for TAG, as was noticed very early, is not the one of derived trees, but the one of derivation trees. This section further explores the modelling power it provides.

In particular, the composition of *the inverse of a function* and a function defines the relation (the "interface") between $\Lambda(\Sigma_{trees})$ and $\Lambda(\Sigma_{logic})$ as $\mathscr{G}_{sem.} \circ \mathscr{G}^{-1}_{derived\ trees}$. In general, such a composition *is not a function*, allowing for relating a derived tree (even more a string) with several logical formulas, and vice versa. This follows the observation of Culicover and Jackendoff (2005) that "[t]he combinatorial principles of syntax and semantics are independent; there is no 'rule-to-rule'

homomorphism. (…) [T]he mapping between syntactic and semantic combinatoriality is many-to-many". However, we implement the many-to-many relation with homomorphisms and inverses of homomorphisms.

In this section, we illustrate the power of this architecture that makes derivation structures a full grammatical object with three phenomena: idioms, subordinating conjunctions with reduced clauses, and scope ambiguity. For idioms, we use the fact that derivation structures are first-class citizens of the formalism. While this could also be expressed in TAG (for instance following the interpretation of derivation trees provided by Shieber 1994), this naturally fits our architecture. For subordinating conjunctions, we rely on the fact that the typing of abstract terms does not need to stick to the tree structure, and in particular to the Gorn addresses, unlike derivation trees in TAG, extending the modelling capabilities. Finally, for scope ambiguity, we show how our approach can take into account analyses from other formalisms, such as categorial and type-logical grammars. We do this remaining in the ACG model, contrary to the TAG extension (TAG with cosubstitution, Barker 2010) to which it corresponds. Other examples that go beyond TAG capabilities are discussed in Section 9, in particular for discourse parsing.

8.1 *Idioms*

Because TAGs provide whole fragments of phrase structures, they can encode the rigid parts of idioms as well as the ones that are subject to possible modifications. Moreover, the role of the derivation structure as a bridge to semantic interpretation nicely captures the relation between a composed syntax and an atomic meaning. With the ACG encoding of TAG, Kobele (2012) shows that we can introduce a constant term that is interpreted as the combination (by adjunction or substitution) of several elementary trees. It goes beyond the previous approaches (Abeillé and Schabes 1989; Shieber and Schabes 1990; Abeillé 1995) in that the derived tree does not need to be an elementary tree of the grammar, but is instead the result of a partial derivation.

We illustrate this with (32):

(32) John kicked the bucket.

Figure 29 presents the initial trees that allow us to analyze (32) as the literal expression, with a compositional meaning built out of the composition of the initial trees α_{John}, α_{kicked}, β_{the}, and α_{bucket}. We actually syntactically analyze the idiomatic expression the same way, except that the combination of α_{kicked}, β_{the}, and α_{bucket} is also considered as the interpretation of the constant $c_{kicked\ the\ bucket}$. We use $c_{kicked\ the\ bucket}$ in the idiomatic derivation in Figure 30(c) to stress that there is no corresponding elementary tree. The ACG abstract term we get really corresponds to this derivation, as the term $C_{30(c)}$ of Equation (33) shows. In both cases, the derived tree is the same (Figure 30(a)). However, the derivation trees differ, as Figure 30(b) and Figure 30(c) show.

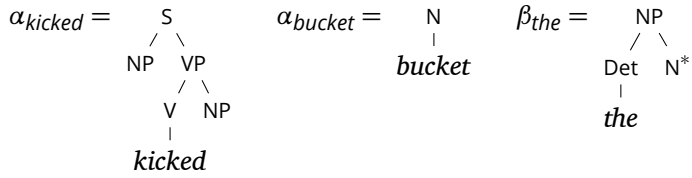
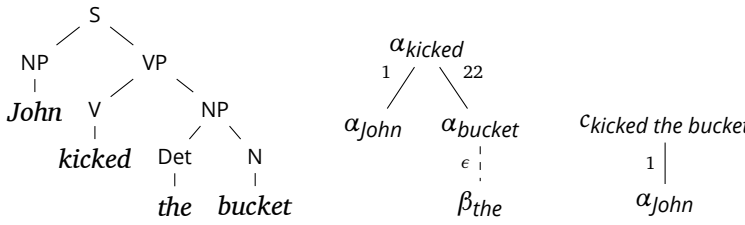


Figure 29: Elementary trees to analyze *kick the bucket*



(a) Derived tree (b) Literal derivation (c) Idiomatic derivation

Figure 30: Derived tree and derivation trees for *John kicked the bucket*

Table 18 (p. 582) shows how the interpretation of the constant abstract term for the idiom is interpreted, syntactically (by $\mathscr{G}_{derived\ trees}$) but not semantically (by $\mathscr{G}_{sem.}$), as the interpretation of the partial derivation $\lambda^{o}s\ a\ subj.c_{kicked}\ s\ a\ subj\ (c_{bucket}\ c_{the})$. Then, according to the lexicons of Tables 17, 18, and 19, (33), (34), (35), and (36) hold. They show that the two terms $C_{30(b)}$ and $C_{30(c)}$ have the same interpretations as derived tree by $\mathscr{G}_{derived\ trees}$. But they have two different interpretations as logical formulas by the ACG $\mathscr{G}_{sem.}$.

(33)
$$C_{30(b)} = C_{kicked}\ I_{S}\ I_{VP}\ C_{John}\ (C_{bucket}\ C_{the})$$
$$C_{30(c)} = C_{kicked\ the\ bucket}\ I_{S}\ I_{VP}\ C_{John}$$

Table 17: TAG elementary tree encoding as $\Lambda(\Sigma_{trees})$ terms

	Terms of $\Lambda(\Sigma_{trees})$	Corresponding TAG elementary tree
γ_{kicked}	$\triangleq \lambda^{\mathbf{o}} S\ a\ s\ o.S\ (\mathsf{S}_2\ s\ (a\ (\mathsf{VP}_2\ (\mathsf{V}_1\ \textbf{\textit{kicked}})\ o)))$ $: (T \multimap T) \multimap (T \multimap T) \multimap T \multimap T \multimap T$	α_{kicked}
γ_{bucket}	$\triangleq \lambda^{\mathbf{o}} d.d\ (\mathsf{N}_1\ \textbf{\textit{bucket}})$ $: (T \multimap T) \multimap T$	α_{bucket}
γ_{the}	$\triangleq \lambda^{\mathbf{o}} n.\mathsf{NP}_2\ (\mathsf{Det}_1\ \textbf{\textit{the}})\ n$ $: T \multimap T$	β_{the}
$\gamma_{kicked\ the\ bucket}$	$\triangleq \lambda^{\mathbf{o}} S\ a\ s.S\ (\mathsf{S}_2\ s\ (a\ (\mathsf{VP}_2\ (\mathsf{V}_1\ \textbf{\textit{kicked}})$ $(\mathsf{NP}_2\ (\mathsf{Det}_1\ \textbf{\textit{the}})(\mathsf{N}_1\ \textbf{\textit{bucket}})))))$ $: (T \multimap T) \multimap (T \multimap T) \multimap T \multimap T$	None: composed from $\alpha_{kicked}, \beta_{the}$, and α_{bucket}

Table 18: Constants of $\Sigma_{derivations}$ and their interpretation by $\mathscr{G}_{derived\ trees}$ and $\mathscr{G}_{sem.}$

c_{kicked}	$: (\mathsf{S} \multimap \mathsf{S}) \multimap (\mathsf{VP} \multimap \mathsf{VP}) \multimap \mathsf{NP} \multimap \mathsf{NP} \multimap \mathsf{S}$
	$:=_{derived\ trees} \gamma_{kicked}$
	$:=_{sem.} \lambda^{\mathbf{o}} s\ a\ \textbf{\textit{subj}}\ \textbf{\textit{obj}}.s\ (\textbf{\textit{subj}}\ (a\ (\lambda x.\textbf{\textit{obj}}\ (\lambda y.\mathbf{kick}\ x\ y))))$
c_{the}	$: \mathsf{N} \multimap \mathsf{NP}$
	$:=_{derived\ trees} \gamma_{the}$
	$:=_{sem.} \lambda^{\mathbf{o}} P\ Q.\exists!\, x.(P\ x) \wedge (Q\ x)$
c_{bucket}	$: (\mathsf{N} \multimap \mathsf{NP}) \multimap \mathsf{NP}$
	$:=_{derived\ trees} \gamma_{bucket}$
	$:=_{sem.} \lambda^{\mathbf{o}} Q.Q\ \mathbf{bucket}$
$c_{kicked\ the\ bucket}$	$: (\mathsf{S} \multimap \mathsf{S}) \multimap (\mathsf{VP} \multimap \mathsf{VP}) \multimap \mathsf{NP} \multimap \mathsf{S}$
	$:=_{derived\ trees} \lambda^{\mathbf{o}} s\ a\ \textbf{\textit{subj}}.(\mathscr{G}_{derived\ trees}(c_{kicked}))\ s\ a\ \textbf{\textit{subj}}$ $((\mathscr{G}_{derived\ trees}(c_{bucket}))\ (\mathscr{G}_{derived\ trees}(c_{the})))$
	$:=_{derived\ trees} \lambda^{\mathbf{o}} s\ a\ \textbf{\textit{subj}}.\gamma_{kicked}\ s\ a\ \textbf{\textit{subj}}\ (\gamma_{bucket}\ \gamma_{the})$
	$:=_{sem.} \lambda^{\mathbf{o}} s\ a\ \textbf{\textit{subj}}.s\ (\textbf{\textit{subj}}\ (a\ (\lambda x.\mathbf{die}\ x)))$

Table 19: Constants of Σ_{TAG} and their interpretation by $\mathscr{G}_{TAG}$

C_{kicked}	$: \mathsf{S}_A \multimap \mathsf{VP}_A \multimap \mathsf{NP} \multimap \mathsf{NP} \multimap \mathsf{S}$
	$:=_{TAG} c_{kicked}$
C_{the}	$: \mathsf{N}_A$
	$:=_{TAG} c_{the}$
C_{bucket}	$\mathsf{N}_A \multimap \mathsf{NP}$
	$:=_{TAG} c_{bucket}$
$C_{kicked\ the\ bucket}$	$\mathsf{S}_A \multimap \mathsf{VP}_A \multimap \mathsf{NP} \multimap \mathsf{S}$
	$:=_{TAG} c_{kicked\ the\ bucket}$

(34)
$$\mathcal{G}_{TAG}(C_{30(b)}) = c_{kicked}\ I_{\mathsf{S}}\ I_{\mathsf{VP}}\ c_{John}\ (c_{bucket}\ c_{the})$$
$$\mathcal{G}_{TAG}(C_{30(c)}) = c_{kicked\ the\ bucket}\ I_{\mathsf{S}}\ I_{\mathsf{VP}}\ c_{John}$$

(35)
$$\mathcal{G}_{derived\ trees} \circ \mathcal{G}_{TAG}(C_{30(b)}) = \mathsf{S}_2\ (\mathsf{NP}_1\ \textit{John})\ (\mathsf{VP}_2\ (\mathsf{V}_1\ \textit{kicked})\ (\mathsf{NP}_2\ (\mathsf{Det}_1\ \textit{the})\ (\mathsf{N}_1\ \textit{bucket})))$$
$$= \mathcal{G}_{derived\ trees} \circ \mathcal{G}_{TAG}(C_{30(c)})$$

(36)
$$\mathcal{G}_{sem.} \circ \mathcal{G}_{TAG}(C_{30(b)}) = \exists!\, x.(\textbf{bucket}\ x) \wedge (\textbf{kick john}\ x)$$
$$\mathcal{G}_{sem.} \circ \mathcal{G}_{TAG}(C_{30(c)}) = \textbf{die john}$$

8.2 *Subordinating conjunctions*

We saw in Section 7.2 that infinitive clauses behave like clauses missing a subject. In this case, the matrix clause (control verb) adjoins on the infinitive clause. As the latter is an argument of the modifier, we could use an extra S'_A type that was interpreted as $(\mathsf{NP} \multimap \mathsf{S}) \multimap \mathsf{S}$ and make the modifier fill the semantic subject with its own subject.

In the case of subordinating conjunctions, as in (37), it is the subordinate clause that adjoins on the matrix clause and uses it as argument, as Figure 31 shows: the substitution node S is meant for the reduced infinitive clause, and the foot node for adjoining into the matrix clause. But if the latter is interpreted as a full proposition, there is no way to *decompose* it so that its subject also fills the semantic subject position of the subordinate clause.

(37) In order to arrive on time, a man left early

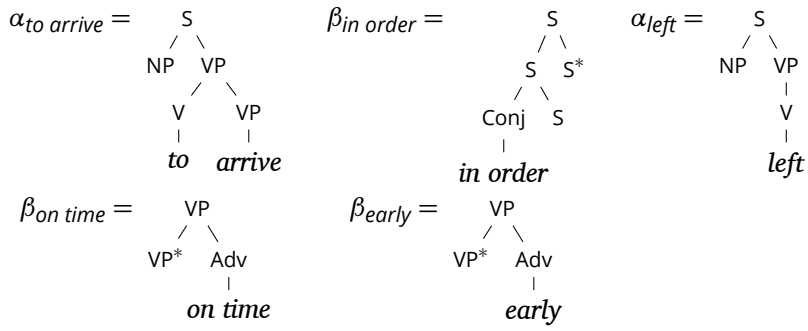


Figure 31: Auxiliary trees for subordinate conjuncts

The solution we propose uses the flexible link between the derivation and the derived trees. The constraints ACGs set on this link have to do with the type, not with the term (provided the typing is preserved). In particular:

- there is no need for an adjunction on a S_n node of a term in $\Lambda(\Sigma_{\textit{trees}})$ to be the image of a term (in $\Lambda(\Sigma_{\textit{derivations}})$) of type S. We already used this feature;
- there is no need for an *actual node* in the derived tree to allow for an adjunction.

In order to implement the solution, terms for verbs such as $c_{\textit{left}}$ in Table 20 have an additional argument of type $((\mathsf{NP} \multimap \mathsf{S}) \multimap (\mathsf{NP} \multimap \mathsf{S}))$ corresponding to the type of the auxiliary trees of subordinate clauses. The latter results for instance from the substitution of an infinitive clause of type $(\mathsf{NP} \multimap \mathsf{S})$ into the term standing for the initial tree of a subordinating conjunction such as $c_{\textit{in order}}$. We can consider this additional argument as an additional possibility to get an adjunction on the S root node (the same node where a $\mathsf{S} \multimap \mathsf{S}$ adjunction is possible). As usual, in case no actual adjunction of a subordinate clause occurs, we use the $I_{\mathsf{NP}\multimap\mathsf{S}}$ constant which is interpreted (syntactically and semantically) as the identity function.

Table 20: Constants of $\Sigma_{\textit{derivations}}$ and their interpretation by $\mathscr{G}_{\textit{derived trees}}$ and $\mathscr{G}_{\textit{sem.}}$

$c_{\textit{left}}$	$: (\mathsf{S} \multimap \mathsf{S}) \multimap ((\mathsf{NP} \multimap \mathsf{S}) \multimap (\mathsf{NP} \multimap \mathsf{S})) \multimap (\mathsf{VP} \multimap \mathsf{VP}) \multimap \mathsf{NP} \multimap \mathsf{S}$
	$:=_{\textit{derived trees}} \gamma_{\textit{left}}$
	$:=_{\textit{sem.}} \lambda^{\mathsf{o}} s\ \textit{sub}\ a\ \textit{subj.sub}\ (\lambda^{\mathsf{o}} \textit{subj}'.s\ (\textit{subj}'\ (a\ (\lambda x.\mathbf{leave}\ x))))\ \textit{subj}$
$c_{\textit{to arrive}}$	$: (\mathsf{VP} \multimap \mathsf{VP}) \multimap \mathsf{NP} \multimap \mathsf{S}$
	$:=_{\textit{derived trees}} \gamma_{\textit{to arrive}}$
	$:=_{\textit{sem.}} \lambda^{\mathsf{o}} \textit{adv}_{\mathsf{VP}}\ \textit{subj.subj}\ (\textit{adv}_{\mathsf{VP}}\ (\lambda x.\mathbf{arrive}\ x))$
$c_{\textit{in order}}$	$: (\mathsf{NP} \multimap \mathsf{S}) \multimap ((\mathsf{NP} \multimap \mathsf{S}) \multimap (\mathsf{NP} \multimap \mathsf{S}))$
	$:=_{\textit{derived trees}} \gamma_{\textit{in order}}$
	$:=_{\textit{sem.}} \lambda^{\mathsf{o}} P\ Q\ \textit{subj.subj}(\lambda x.\mathbf{goal}\ (P\ (\lambda^{\mathsf{o}} p.p\ x))\ (Q\ (\lambda^{\mathsf{o}} p.p\ x)))$
$I_{\mathsf{NP}\multimap\mathsf{S}}$	$: (\mathsf{NP} \multimap \mathsf{S}) \multimap (\mathsf{NP} \multimap \mathsf{S})$
	$:=_{\textit{derived trees}} \lambda^{\mathsf{o}} x.x$
	$:=_{\textit{sem.}} \lambda^{\mathsf{o}} x.x$
$c_{\textit{early}}$	$: \mathsf{VP} \multimap \mathsf{VP}$
	$:=_{\textit{derived trees}} \lambda^{\mathsf{o}} x.\mathsf{VP}_2\ x\ (\mathsf{Adv}_1\ \textit{early})$
	$:=_{\textit{sem.}} \lambda^{\mathsf{o}} p.\lambda x.\mathbf{early}(p\ x)$
$c_{\textit{on time}}$	$: \mathsf{VP} \multimap \mathsf{VP}$
	$:=_{\textit{derived trees}} \lambda^{\mathsf{o}} x.\mathsf{VP}_2\ x\ (\mathsf{Adv}_1\ \textit{on time})$
	$:=_{\textit{sem.}} \lambda^{\mathsf{o}} p.\lambda x.\mathbf{on\_time}(p\ x)$

We can observe in $\mathscr{G}_{derived\ trees}(c_{left}) = \gamma_{left}$ how the subordinate clause is inserted. The latter corresponds to the *sub* argument in γ_{left} in Table 22 (p. 586). It takes as an argument the whole S rooted subtree over which the NP subject is abstracted (with $\lambda^o s'$) and the actual subject *subj* of the matrix clause. So it is the subordinate clause that is responsible for first applying the matrix clause to its subject before plugging in the resulting tree at the foot node. We can observe this behavior in $\gamma_{in\ order}$: the *sub* argument corresponds to the infinitive subordinate clause to be substituted in $\beta_{in\ order}$, while the *matrix* argument corresponds to the matrix clause into which it adjoins and to which the *subj* argument is given, as the subterm (*matrix subj*) shows.

As before, the higher-order types at the level of $\Sigma_{derivations}$ are interpretations of atomic types of Σ_{TAG}. In particular, we introduce the atomic type $\mathsf{S}''_A :=_{TAG} (\mathsf{NP} \multimap \mathsf{S}) \multimap (\mathsf{NP} \multimap \mathsf{S})$ (resp. $\mathsf{S}_{\mathsf{WS}} :=_{TAG} \mathsf{NP} \multimap \mathsf{S}$) for the reduced subordinate clauses (resp. for the infinitive clause that occurs in subordinate clauses) as Table 21 shows.

S''_A	$:=_{TAG} (\mathsf{NP} \multimap \mathsf{S}) \multimap (\mathsf{NP} \multimap \mathsf{S})$
S_{WS}	$:=_{TAG} \mathsf{NP} \multimap \mathsf{S}$
C_{left}	$: \mathsf{S}_A \multimap \mathsf{S}''_A \multimap \mathsf{VP}_A \multimap \mathsf{NP} \multimap \mathsf{S}$
	$:=_{TAG} c_{left}$
$C_{to\ arrive}$	$: \mathsf{VP}_A \multimap \mathsf{S}_{\mathsf{WS}}$
	$:=_{TAG} c_{to\ arrive}$
$C_{in\ order}$	$: \mathsf{S}_{\mathsf{WS}} \multimap \mathsf{S}''_A$
	$:=_{TAG} c_{in\ order}$
$I_{\mathsf{S}''}$	$: \mathsf{S}''_A$
	$:=_{TAG} I_{\mathsf{NP} \multimap \mathsf{S}}$
C_{early}	$: \mathsf{VP}_A$
	$:=_{TAG} c_{early}$
$C_{on\ time}$	$: \mathsf{VP}_A$
	$:=_{TAG} c_{on\ time}$

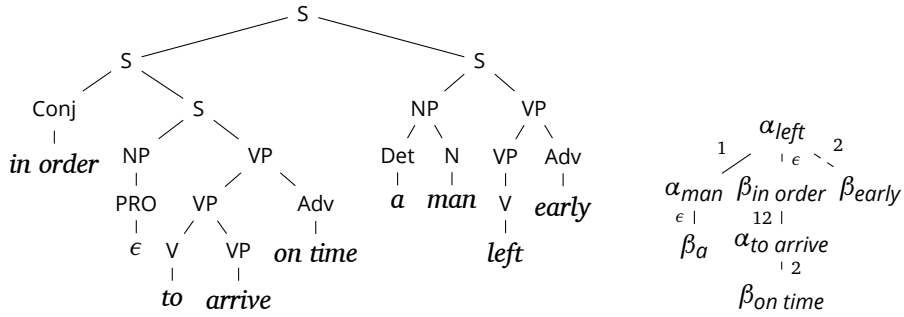
Table 21:
Constants of Σ_{TAG}
and their interpretation by $\mathscr{G}_{TAG}$

With the lexicon of Tables 20, 21, and 22, we can build terms that correspond to the derivation and derived trees of Figure 32 as (38), (39), and (40) show.[26] We compute the semantic interpretation as in (41).

[26] Note that all terms corresponding to initial trees where the adjunction of a subordinate clause can occur should have the extra argument added. For the sake of simplicity, only C_{left} and c_{left} are modified here.

Table 22: TAG elementary tree encoding as $\Lambda(\Sigma_{trees})$ terms

Terms of $\Lambda(\Sigma_{trees})$	Corresponding TAG elementary tree
$\gamma_{left} = \lambda^{o} S\ sub\ a\ subj\ obj.$ $S\ (sub\ (\lambda^{o} s'.\mathsf{S}_2\ s'\ (a\ (\mathsf{VP}_1\ (\mathsf{V}_1\ left))))\ subj)$ $: (T \multimap T) \multimap ((T \multimap T) \multimap (T \multimap T)) \multimap (T \multimap T) \multimap T \multimap T \multimap T$	α_{left}
$\gamma_{to\ arrive} = \lambda^{o} a\ s.\mathsf{S}_2\ (\mathsf{NP}_1\ s)\ (a\ (\mathsf{VP}_2\ (\mathsf{V}_1\ to)\ (\mathsf{VP}_1\ arrive)))$ $: (T \multimap T) \multimap T \multimap T$	$\alpha_{to\ arrive}$
$\gamma_{in\ order} = \lambda^{o} sub\ matrix\ subj.$ $\mathsf{S}_2\ (\mathsf{S}_2\ (\mathsf{Conj}_1\ in\ order)\ (sub\ (\mathsf{PRO}_1\ \epsilon)))\ (matrix\ subj)$ $: (T \multimap T) \multimap (T \multimap T) \multimap T \multimap T$	$\beta_{in\ order}$



(a) Derived tree (b) Derivation tree

Figure 32: Derived tree and derivation tree for *In order to arrive on time, a man left early*

(38) $C_{32} = C_{left}\ I_{\mathsf{S}}\ (C_{in\ order}\ (C_{to\ arrive}\ C_{on\ time}))\ C_{early}\ (C_{man}\ C_a)$

(39) $\mathscr{G}_{TAG}(C_{32}) = c_{left}\ I_{\mathsf{S}}\ (c_{in\ order}\ (c_{to\ arrive}\ c_{on\ time}))\ c_{early}\ (c_{man}\ c_a)$

(40) $\mathscr{G}_{derived\ trees} \circ \mathscr{G}_{TAG}(C_{32}) =$
$\mathsf{S}_2\ (\mathsf{S}_2\ (\mathsf{Conj}_1\ in\ order)\ (\mathsf{S}_2\ (\mathsf{NP}_1\ (\mathsf{PRO}_1\ \epsilon))(\mathsf{VP}_2$
$(\mathsf{VP}_2\ (\mathsf{V}_1 to)\ (\mathsf{VP}_1\ arrive))(\mathsf{Adv}_1\ on\ time))))$
$(\mathsf{S}_2\ (\mathsf{NP}_2\ (\mathsf{Det}_1\ a)\ (\mathsf{N}_1\ man))$
$(\mathsf{VP}_2\ (\mathsf{VP}_1\ (\mathsf{V}_1\ left))\ (\mathsf{Adv}_1\ early)))$

(41) $\mathcal{G}_{sem.} \circ \mathcal{G}_{TAG}(C_{32}) = \exists x.(\mathbf{man}\ x) \wedge (\mathbf{goal}(\mathbf{on\_time}(\mathbf{arrive}\ x))(\mathbf{early}(\mathbf{leave}\ x)))$

Because $\mathcal{G}_{TAG}$ is still second-order, parsing is available. Parsing the logical term t_{32}^{logic} (see (42)) results in the term $t_{32} : \mathsf{s}$ of $\mathcal{A}(\mathcal{G}_{sem.} \circ \mathcal{G}_{TAG})$. This is the same term of $\mathcal{A}(\mathcal{G}_{yield} \circ \mathcal{G}_{derived\ trees} \circ \mathcal{G}_{TAG})$ that we get when parsing t_{32}^{string}.

(42) $t_{32}^{\text{logic}} = \exists x.(\mathbf{man}\ x) \wedge (\mathbf{goal}(\mathbf{on\_time}(\mathbf{arrive}\ x))(\mathbf{early}(\mathbf{leave}\ x)))$

(43) $t_{32}^{\text{string}} = in + order + to + arrive + on + time + a + man + left + early$

(44) $t_{32} = C_{left}\ I_{\mathsf{s}}\ (C_{in\ order}\ (C_{to\ arrive}\ C_{on\ time}))\ C_{early}\ (C_{man}\ C_{a})$

8.3 *Scope ambiguity and non-functional form-meaning relation*

The phenomena we have modelled so far make use of derivation structures (either in $\Lambda(\Sigma_{derivations})$ or in $\Lambda(\Sigma_{TAG})$) that are very close (homomorphic) to TAG derivation trees. As we can see in Figure 15 (p. 557), the relation between TAG derivations as terms of $\Lambda(\Sigma_{TAG})$ and terms of $\Lambda(\Sigma_{logic})$ is functional (encoded by the composition $\mathcal{G}_{sem.} \circ \mathcal{G}_{TAG}$). The non-functional relation is between terms of $\Lambda(\Sigma_{trees})$ and terms of $\Lambda(\Sigma_{logic})$ (encoded by the relation $\mathcal{G}_{sem.} \circ \mathcal{G}_{derived\ trees}^{-1}$). So because there are two derivation trees for *John kicked the bucket*, there are two possible semantic interpretations. But with only one derivation tree for *every man loves some woman*, there is only one possible semantic interpretation. A possible solution to this problem is to use an underspecified representation formalism instead of higher-order logic to represent the semantics, as Pogodalla (2004a) proposes.

We present here another solution. It uses the power of higher-order typing of the abstract terms in order to provide TAGs with a relation between TAG derivation trees and meanings that *is not functional*. Nevertheless, our grammatical architecture only appeals to homomorphisms. We introduce an abstract vocabulary Σ_{CoTAG} and two ACGs. The first one, $\mathcal{G}_{CoTAG}$, maps terms of $\Lambda(\Sigma_{CoTAG})$ to terms of $\Lambda(\Sigma_{TAG})$, i.e., TAG derivation trees. The second one, $\mathcal{G}_{co\text{-}sem.}$, maps terms of $\Lambda(\Sigma_{CoTAG})$ to terms of $\Lambda(\Sigma_{logic})$. It then provides a relation between $\Lambda(\Sigma_{TAG})$ and $\Lambda(\Sigma_{logic})$ as $\mathcal{G}_{co\text{-}sem.} \circ \mathcal{G}_{CoTAG}^{-1}$ as Figure 33 (p. 588) shows. The derivation tree (in $\Lambda(\Sigma_{TAG})$) of *every man loves some woman*, for

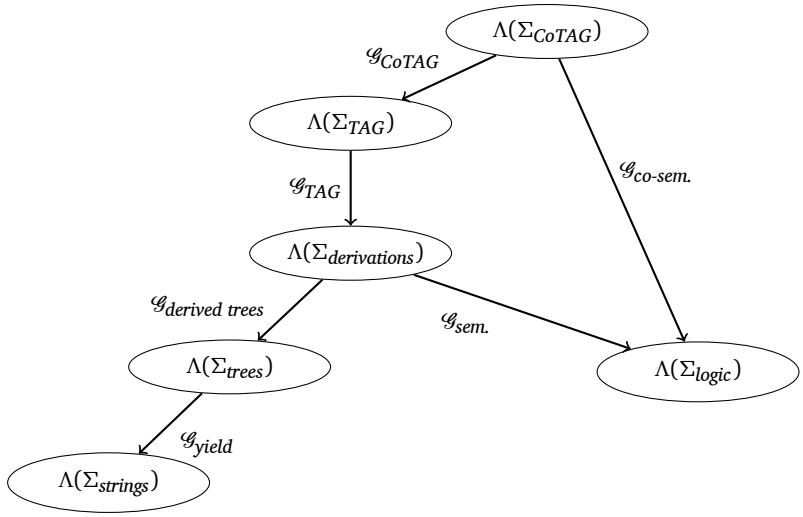


Figure 33: ACG composition for TAG and CoTAG

L_{John}	$: \mathsf{NP}$	$\coloneqq_{CoTAG} C_{John}$
L_{loves}	$: \mathsf{S}_A \multimap \mathsf{VP}_A \multimap \mathsf{NP} \multimap \mathsf{NPS}$	$\coloneqq_{CoTAG} C_{loves}$
$L_{everyone}$	$: (\mathsf{NP} \multimap \mathsf{S}) \multimap \mathsf{S}$	$\coloneqq_{CoTAG} \lambda^{\mathrm{o}} P.P\ C_{everyone}$
$L_{someone}$	$: (\mathsf{NP} \multimap \mathsf{S}) \multimap \mathsf{S}$	$\coloneqq_{CoTAG} \lambda^{\mathrm{o}} P.P\ C_{someone}$
L_{every}	$: \mathsf{N}_A$	$\coloneqq_{CoTAG} C_{every}$
L_{some}	$: \mathsf{N}_A$	$\coloneqq_{CoTAG} C_{some}$
L_{man}	$: \mathsf{N}_A \multimap (\mathsf{NP} \multimap \mathsf{S}) \multimap \mathsf{S}$	$\coloneqq_{CoTAG} \lambda^{\mathrm{o}} det\ P.P\ (C_{man}\ det)$
L_{woman}	$: \mathsf{N}_A \multimap (\mathsf{NP} \multimap \mathsf{S}) \multimap \mathsf{S}$	$\coloneqq_{CoTAG} \lambda^{\mathrm{o}} det\ P.P\ (C_{woman}\ det)$

Table 23: Constants of Σ_{CoTAG} and their interpretation by $\mathcal{G}_{CoTAG}$

instance, will have *two* antecedents in $\Lambda(\Sigma_{CoTAG})$, hence two semantic interpretations.

The idea is to use the type-raising methods of categorial and type-logical grammars. So, corresponding to a term $C_{everyone} : \mathsf{NP}$ in Σ_{TAG}, we have a term $L_{everyone} : (\mathsf{NP} \multimap \mathsf{S}) \multimap \mathsf{S}$ in Σ_{CoTAG} such that $\mathcal{G}_{CoTAG}(L_{everyone}) = \lambda^{\mathrm{o}} P.P\ C_{everyone}$. More generally, whenever a term of type A occurring within a constituent of type B can take scope over this term, we associate to $C_{scoping} : A_1 \multimap \ldots \multimap A_n \multimap A$ in Σ_{TAG} a term $L_{scoping} : A_1 \multimap \ldots \multimap A_n \multimap (A \multimap B) \multimap B$ in Σ_{CoTAG} such that $\mathcal{G}_{CoTAG}(L_{scoping}) = \lambda^{\mathrm{o}} x_1 \ldots x_n . \lambda^{\mathrm{o}} P.P\ (C_{scoping}\ x_1 \cdots x_n)$. For other lexical items $C_{lex.\ item} : \alpha$, we have $L_{lex.item} : \alpha$ such that $\mathcal{G}_{CoTAG}(L_{lex.item}) = C_{lex.\ item}$. And for any atomic type A, $\mathcal{G}_{CoTAG}(A) = A$. Table 23 exemplifies the approach for quantified noun phrases (note that proper

nouns, for instance, are not type-raised). Atomic types in Σ_{CoTAG} are the same as in Σ_{TAG}. With L_{21}^{sws} and L_{21}^{ows} as defined in (45) and (46) respectively, we indeed have $\mathscr{G}_{CoTAG}(L_{21}^{sws}) = \mathscr{G}_{CoTAG}(L_{21}^{ows})$, i.e., two different abstract terms of $\Lambda(\Sigma_{CoTAG})$ that are mapped to the same term of $\Lambda(\Sigma_{TAG})$ (TAG derivation tree).

(45)

$$\begin{aligned} L_{21}^{sws} &= (L_{man}\ L_{every}) \\ &\qquad (\lambda^{o}x.(L_{woman}\ L_{some})\ (\lambda^{o}y.L_{loves}\ I_{\mathsf{S}}\ I_{\mathsf{VP}}\ x\ y)) \\ \mathscr{G}_{CoTAG}(L_{21}^{sws}) &= (\lambda^{o}P.P\ (C_{man}\ C_{every}))\ (\lambda^{o}x.(\lambda^{o}P.P\ (C_{woman}\ C_{some})) \\ &\qquad (\lambda^{o}y.C_{loves}\ I_{\mathsf{S}}\ I_{\mathsf{VP}}\ x\ y)) \\ &= C_{loves}\ I_{\mathsf{S}}\ I_{\mathsf{VP}}\ (C_{man}\ C_{every})\ (C_{woman}\ C_{some}) \end{aligned}$$

(46)

$$\begin{aligned} L_{21}^{ows} &= (L_{woman}\ L_{some}) \\ &\qquad (\lambda^{o}y.(L_{man}\ L_{every})\ (\lambda^{o}x.L_{loves}\ I_{\mathsf{S}}\ I_{\mathsf{VP}}\ x\ y)) \\ \mathscr{G}_{CoTAG}(L_{21}^{ows}) &= (\lambda^{o}P.P\ (C_{woman}\ C_{some}))\ (\lambda^{o}y.(\lambda^{o}P.P\ (C_{man}\ C_{every})) \\ &\qquad (\lambda^{o}x.C_{loves}\ I_{\mathsf{S}}\ I_{\mathsf{VP}}\ x\ y)) \\ &= C_{loves}\ I_{\mathsf{S}}\ I_{\mathsf{VP}}\ (C_{man}\ C_{every})\ (C_{woman}\ C_{some}) \end{aligned}$$

In order to get two semantic interpretations from the two abstract terms of $\Lambda(\Sigma_{CoTAG})$, we need to directly provide them with a semantic lexicon. For if we keep on interpreting them through Σ_{TAG} and $\Sigma_{derivations}$, because the two terms L_{21}^{sws} and L_{21}^{ows} are interpreted as a single term in $\Lambda(\Sigma_{TAG})$, we would still get a single interpretation. In other words, we do not want the diagram of Figure 33 to commute.

Table 24 defines the $\mathscr{G}_{co\text{-}sem.}$ interpretation into terms of $\Lambda(\Sigma_{logic})$. Contrary to $\mathscr{G}_{sem.}$ where NPs are interpreted with the higher-order type $(e \to t) \multimap t$, because quantified noun phrases are given the type $(\mathsf{NP} \multimap \mathsf{S}) \multimap \mathsf{S}$ in Σ_{CoTAG}, we now interpret NP as e. All the other interpretations, in particular for verbs, are defined accordingly.[27]

We can now compute the semantic interpretation of L_{21}^{sws} and L_{21}^{ows} by $\mathscr{G}_{co\text{-}sem.}$. Equations (47) and (48) show that these two terms are

[27] Note, however, that, because at the abstract level we only have linear types, in order to allow for non linearity at the object level, we have to uniformly interpret $\multimap$ as $\to$, so that the image of $(\mathsf{NP} \multimap \mathsf{S}) \multimap \mathsf{S}$ is $(e \to t) \to t$. Another possibility would be to use the exponential connectives of linear logic.

Table 24: Constants of Σ_{CoTAG} and their interpretation by $\mathscr{G}_{co\text{-}sem.}$

NP	$:=_{co\text{-}sem.} e$
VP	$:=_{co\text{-}sem.} e \to t$
L_{John}	$:=_{co\text{-}sem.} \mathbf{john}$
L_{loves}	$:=_{co\text{-}sem.} \lambda adv_s adv_{VP} subj obj.adv_s\ (adv_{VP}\ (\lambda x.\mathbf{love}\ x\ obj)\ subj)$
$L_{everyone}$	$:=_{co\text{-}sem.} \lambda Q.\forall x.(\mathbf{human}\ x) \Rightarrow (Q\ x)$
$L_{someone}$	$:=_{co\text{-}sem.} \lambda Q.\exists x.(\mathbf{human}\ x) \wedge (Q\ x)$
L_{every}	$:=_{co\text{-}sem.} \lambda P\ Q.\forall x.(P\ x) \Rightarrow (Q\ x)$
L_{some}	$:=_{co\text{-}sem.} \lambda P\ Q.\exists x.(P\ x) \wedge (Q\ x)$
L_{man}	$:=_{co\text{-}sem.} \lambda det.det\ \mathbf{man}$
L_{woman}	$:=_{co\text{-}sem.} \lambda det.det\ \mathbf{woman}$

mapped onto two logical formulas corresponding to the subject wide scope reading on the one hand, and to the object wide scope reading on the other hand.

(47) $\mathscr{G}_{co\text{-}sem.}(L_{21}^{sws}) = \forall x.(\mathbf{man}\ x) \Rightarrow (\exists x'.(\mathbf{woman}\ x') \wedge (\mathbf{love}\ x\ x'))$

(48) $\mathscr{G}_{co\text{-}sem.}(L_{21}^{ows}) = \exists x.(\mathbf{woman}\ x) \wedge (\forall x'.(\mathbf{man}\ x') \Rightarrow (\mathbf{love}\ x'\ x))$

This approach to scope ambiguity, first proposed by Pogodalla (2007b,a), is used by Kobele and Michaelis (2012) to provide an ACG formalization of the cosubstitution operation for TAG (Barker 2010). This also makes explicit Barker's (2010) claim that "cosubstitution is a version of the continuation-based approaches to scope-taking […]". And, indeed, the type $(\mathsf{NP} \multimap \mathsf{S}) \multimap \mathsf{S}$ corresponds to making the continuation of a noun phrase (i.e., its scope) part of its interpretation.

$\mathscr{G}_{CoTAG}$ is not a second-order ACG. In this particular case, because of lexicalization, we know that parsing is decidable, but it can be complex. Salvati (2007) presents a lexicalized third-order ACG whose membership problem reduces to an NP-complete problem. There is currently no implementation of parsing for such grammars in the ACG toolkit. The identification of fragments that are both linguistically relevant and computationally tractable is ongoing work.

This extension with more abstract levels can also be used to model (non-local) MCTAG. And one level more can control MCTAG (similar to the control that $\mathscr{G}_{TAG}$ adds on $\mathscr{A}(\mathscr{G}_{derived\ trees})$) so that it stays within the polynomially parsable languages of set-local MCTAG (Pogodalla 2009).

9 RELATED APPROACHES

Moving to ACGs to encode TAGs and to build TAG semantic representations offers several advantages. First, we saw in Section 3.2 that we can benefit from parsing algorithms and optimization techniques grounded in well-established fields such as type theory and Datalog.

A second advantage, also concerning the parsing algorithms, is to offer an *inherently reversible* framework (Dymetman 1994). Kanazawa's Datalog reduction (2007; 2017) indeed makes no hypothesis on the object language: it can be a language of strings, of trees, or of any kind of (almost linear) λ-terms. In the latter case, it can represent the usual logical semantic formulas. While in NLP *parsing* usually refers to building a parse structure (or a semantic term) from a string representation and *generation* (or *surface realization*) refers to building a string from a semantic representation, they both rely on ACG parsing (i.e., recovering the abstract structure from an object term), and the algorithms are the same.

This constitutes an important difference between the ACG approach and the synchronous approaches to semantic construction. If both are based on (or can be reformulated using) a tree transduction, the latter does not offer a built-in transformation to β-reduced terms (which may definitely not be trees but rather graphs) at the semantic level. When parsing strings, synchronous grammars (Nesson and Shieber 2006; Nesson 2009) build semantic trees that correspond to λ-terms *before* the β-reduction. Processing such trees to produce the β-reduced form is straightforward. However the inverse process, the one that is of interest for generation, is not. It actually corresponds to the morphism inversion found in ACG parsing. The ACG framework tells us that this inversion is possible, and, for the second-order case, it has actually been implemented.

Koller and Kuhlmann (2011, 2012) also propose parsing by morphism inversion using interpreted regular tree grammars, and their approach completely fits the synchronous approach. But, as for synchronous TAG, this formalism is not well-suited to dealing with semantics represented with logical formulas. To parse a term t requires that the set of trees that are interpreted as t is regular. For instance, if the string algebra comes with the 2-ary concatenation operation, this

set is the set of all the bracketings of the string to parse (Koller and Kuhlmann 2011) (the string algebra Koller and Kuhlmann 2012 propose for the TAG encoding is different, in order to keep the complexity bound for parsing low). Applying the same approach to logical representations based on λ-calculus would mean representing all terms that are β-equivalent to the term we want to parse by a regular tree grammar. It is not clear how this can be done.

Semantic representation with λ-terms, and the ACG type-theoretic settings more generally,[28] also provides tight links with formal logical semantics. The various grammatical formalisms ACGs can encode may be linked to various semantic theories. This concerns both semantic theories, such as event semantics (Davidson 2001) (for a type-theoretic account, see Blom *et al.* 2012) or dynamic semantics (Kamp and Reyle 1993; Groenendijk and Stokhof 1991) (for a type-theoretic semantics account, see de Groote 2006; Martin and Pollard 2014),[29] and phenomena at the syntax-semantics interface where approaches based on underspecification (Pogodalla 2004a,b) or based on type theory and higher-order logic (Pogodalla 2007b,a; Kobele and Michaelis 2012) can be expressed.

Because they only use unification, the unification-based approaches to TAG semantics (Gardent and Kallmeyer 2003; Kallmeyer and Romero 2004, 2008) do not easily extend to higher-order semantics: only conjunctions of propositions are allowed, and no application. A first consequence is that the actual representation language needs to be embedded into a reified logical language (typically a labelled underspecified representation language). For instance, the semantics of an adverb adjoining to a VP node cannot be represented as a function from $(e \to t)$ to $(e \to t)$. It is represented as a proposition expressing that some property holds of a label which gets its value by unification with the label corresponding to the semantics of the VP in the verb initial tree. When dealing with higher-order representations, as for dynamic semantics of discourse (de Groote 2006; Martin and Pollard

[28] This includes other categorial grammars (van Benthem 1986; Carpenter 1997; Steedman 2001; Steedman and Baldridge 2011).

[29] Note however that the semantic calculi are somewhat extended with additional operators and then do not fulfill the requirements allowing for reversibility. This is a research program on its own.

2014), it becomes awkward to assign values of arguments with unification and to compute the semantic representation by β-reduction.

Moreover, ACGs uniformly deal with the interpretation of derivation trees, either as strings, derived trees, or semantic representations. Consequently, the same parsing algorithms apply. This is not the case for the unification-based approaches, and the reversibility of the grammars is not ensured.

Another benefit of the ACG approach to the syntax-semantics interface over the synchronous TAG or over the unification-based approach is that, by construction, it is compositional, and the homomorphism requirement between the syntactic and the semantic categories holds. For instance, in synchronous TAG, in a pair of syntactic and semantic trees, it is possible to link a node X (in the syntactic tree) with a node of type α (in the semantic tree), while having a pair of auxiliary trees whose syntactic tree has a foot and a root node labelled by X, but the nodes in the semantic trees are labelled by $\beta \neq \alpha$, yielding semantic trees that are not well-typed. A similar thing can happen in unification-based approaches if the semantic features to be unified are not the same. This is not possible in ACG and the ACG toolkit would raise a typing error, in the same way that statically typed programming languages ensure type-safeness.

Finally, the modularity of the ACG framework allows us to look at TAG and TAG variants as fragments of a larger class of grammars. For instance, Multi-Component TAG (MCTAG: Weir 1988) can also be described using a similar architecture (Pogodalla 2009). It is also possible to add operations in addition to substitution and adjunction that would otherwise be difficult, if not impossible, to express as TAG (or MCTAG) operations. Such operations can be used in order to link a TAG phrase grammar with a TAG discourse grammar without requiring an intermediate processing step (Danlos *et al.* 2015, 2016), contrary to D-LTAG (Webber and Joshi 1998; Forbes *et al.* 2003; Webber 2004; Forbes-Riley *et al.* 2006) or D-STAG (Danlos 2009, 2011). But, provided the encoding remains in the second-order ACG class, these grammars remain reversible and there is no need to design new parsing algorithms.

10 CONCLUSION

We have presented a model of the syntax-semantics interface for TAGs hinging on the ACG framework. We demonstrated, with the help of classical TAG syntax-semantics examples and new modellings, that this framework offers a lot of flexibility and expressiveness. In particular, we built on the modular properties of ACG that result from the two notions of composition between grammars it provides. These composition modes have been used for the syntax-semantics interface on the one hand, and for restricting the derivations to actual TAG derivations using a second-order ACG on the other hand. This allowed us to apply the ACG parsing results and to make the grammar reversible so that both parsing and syntactic realization are available.

Moreover, we showed that new modellings can be proposed that extend the standard TAG analyses without the requirement of designing new parsing algorithms. This was illustrated with phenomena such as idioms and subordinating conjunctions. We also showed what we can bring into TAG accounts from type-logical frameworks, such as the modelling of scope ambiguities.

This shows how relevant ACGs are as models of the syntax-semantics interface in general, and for TAG in particular. Relying on the work we have presented here, we can consider modelling other standard extensions of TAGs, such as MCTAG. We can also consider relating TAG to other type-theoretic modellings of semantic phenomena, e.g., discourse, knowledge and beliefs, time, etc. Finally, we believe this can give a new perspective on ways to model phenomena which are challenging to model otherwise in TAG, such as coordination.

REFERENCES

Anne Abeillé (1990), French and English determiners: interaction of morphology, syntax and semantics in Lexicalized Tree Adjoining Grammars, in Karin Harbusch and Wolfgang Wahlster, editors, *Proceedings of the 1st International Workshop on Tree Adjoining Grammars: Formal Theory and Applications*, pp. 17–20, Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloß Dagstuhl, Germany, ACL anthology: W90-0215.

Anne Abeillé (1993), *Les nouvelles syntaxes*, Armand Colin.

Anne Abeillé (1995), The flexibility of French idioms: a representation with Lexicalised Tree Adjoining Grammar, in Martin Everaert, Erik-Jan van der

LINDEN, André SCHENK, and Rob SCHREUDER, editors, *Idioms: structural and psychological perspectives*, chapter 1, pp. 15–42, Psychology Press, Taylor & Francis Group.

Anne ABEILLÉ (2002), *Une grammaire électronique du français*, Sciences du langage, CNRS Éditions.

Anne ABEILLÉ and Yves SCHABES (1989), Parsing idioms in Lexicalized TAGs, in *Proceedings of the Fourth Conference of the European Chapter of the Association for Computational Linguistics (EACL 1989)*, pp. 1–9, Association for Computational Linguistics, Manchester, England, ACL anthology: E89-1001.

Hendrik Pieter BARENDREGT (1984), *The lambda calculus: its syntax and semantics*, volume 103 of *Studies in logic and the foundations of mathematics*, North-Holland.

Chris BARKER (2010), Cosubstitution, derivational locality, and quantifier scope, in Srinivas BANGALORE, Robert FRANK, and Maribel ROMERO, editors, *Proceedings of the 10th International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG + 10)*, pp. 135–142, Linguistics Department, Yale University, New Haven, CT, USA, ACL anthology: W10-4417.

Katalin BIMBÓ (2015), The decidability of the intensional fragment of classical linear logic, *Theoretical Computer Science*, 597:1–17, 10.1016/j.tcs.2015.06.019.

Philippe BLACHE, Edward STABLER, Joan BUSQUETS, and Richard MOOT, editors (2005), *Proceedings of the 5th international conference on Logical Aspects of Computational Linguistics (LACL 2005)*, volume 3492 of *Lecture notes in computer science/Lecture notes in artificial intelligence*, Springer, 10.1007/b136076.

Chris BLOM, Philippe DE GROOTE, Yoad WINTER, and Joost ZWARTS (2012), Implicit arguments: event modification or option type categories?, in Maria ALONI, Vadim KIMMELMAN, Floris ROELOFSEN, Galit W. SASSOON, Katrin SCHULZ, and Matthijs WESTERA, editors, *Logic, language and meaning*, volume 7218 of *Lecture notes in computer science*, pp. 240–250, Springer, 10.1007/978-3-642-31482-7_25.

Johan BOS (1995), Predicate logic unplugged, in Paul DEKKER and Martin STOKHOF, editors, *Proceedings of the Tenth Amsterdam Colloquium*, ILLC, University of Amsterdam, `http://www.let.rug.nl/bos/pubs/Bos1996AmCo.pdf`.

Pierre BOURREAU (2012), *Jeux de typage et analyse de λ-grammaires non-contextuelles*, Ph.D. thesis, Université Bordeaux I, HAL open archive: tel-00733964.

Pierre BOURREAU (2013), Traitements d'ellipses: deux approches par les grammaires catégorielles abstraites, in *Actes de la 20^e conférence sur le Traitement Automatique des Langues Naturelles (TALN 2013)*, pp. 215–228, Association pour le Traitement Automatique des Langues, Les Sables d'Olonne, France, `http://talnarchives.atala.org/TALN/TALN-2013/taln-2013-long-016.pdf`.

Pierre BOURREAU and Sylvain SALVATI (2011), A Datalog recognizer for almost affine λ-CFGs, in Makoto KANAZAWA, András KORNAI, Marcus KRACHT, and Hiroyuki SEKI, editors, *The mathematics of language*, volume 6878 of *Lecture notes in computer science*, pp. 21–38, Springer, 10.1007/978-3-642-23211-4_2.

Marie-Hélène CANDITO (1996), A principle-based hierarchical representation of LTAGs, in *Proceedings of the 16th International Conference on Computational Linguistics (COLING 1996)*, pp. 194–199, ACL anthology: C96-1034.

Marie-Hélène CANDITO (1999), *Représentation modulaire et paramétrable de grammaires électroniques lexicalisées: application au français et à l'italien*, Ph.D. thesis, Université Paris 7, `http://www.linguist.univ-paris-diderot.fr/~mcandito/Publications/candito-these.pdf`.

Marie-Hélène CANDITO and Sylvain KAHANE (1998), Can the TAG derivation tree represent a semantic graph? An answer in the light of Meaning-Text Theory, in Anne ABEILLÉ, Tilman BECKER, Owen RAMBOW, Giorgio SATTA, and K. VIJAY-SHANKER, editors, *Proceedings of the Fourth International Workshop on Tree Adjoining Grammars and Related Frameworks (TAG+4)*, volume 98-12 of *IRCS Report*, University of Pennsylvania, ACL anthology: W98-0106.

Bob CARPENTER (1997), *Type-logical semantics*, The MIT Press.

John CHEN, Srinivas BANGALORE, and K. VIJAY-SHANKER (2006), Automated extraction of Tree-Adjoining Grammars from treebanks, *Natural Language Engineering*, 12(3):251–299, 10.1017/S1351324905003943.

Hubert COMON, Max DAUCHET, Rémi GILLERON, Christof LÖDING, Florent JACQUEMARD, Denis LUGIEZ, Sophie TISON, and Marc TOMMASI (2007), Tree Automata techniques and applications, `http://www.grappa.univ-lille3.fr/tata`, released October 12th, 2007.

Benoît CRABBÉ (2005), Grammatical development with XMG, in Blache *et al.* (2005), pp. 84–100, 10.1007/11422532_6.

Benoît CRABBÉ, Denys DUCHIER, Claire GARDENT, Joseph Le ROUX, and Yannick PARMENTIER (2013), XMG: eXtensible MetaGrammar, *Computational Linguistics*, 39(3):591–629, ACL anthology: J13-3005.

Peter W. CULICOVER and Ray JACKENDOFF (2005), *Simpler syntax*, Oxford University Press.

Haskell Brooks CURRY (1961), Some logical aspects of grammatical structure, in Roman JAKOBSON, editor, *Structure of language and its mathematical aspects: proceedings of the twelfth symposium in applied mathematics*, pp. 56–68, American Mathematical Society.

Laurence DANLOS (2009), D-STAG: un formalisme d'analyse automatique de discours basé sur les TAG synchrones, *Revue TAL*, 50(1):111–143, HAL open archive: inria-00524743.

Laurence DANLOS (2011), D-STAG: a formalism for discourse analysis based on SDRT and using Synchronous TAG, in Philippe DE GROOTE, Markus EGG, and Laura KALLMEYER, editors, *Proceedings of the 14th conference on Formal Grammar (FG 2009)*, volume 5591 of *Lecture notes in computer science/Lecture notes in artificial intelligence*, pp. 64–84, Springer, 10.1007/978-3-642-20169-1_5.

Laurence DANLOS, Aleksandre MASKHARASHVILI, and Sylvain POGODALLA (2015), Grammaires phrastiques et discursives fondées sur les TAG: une approche de D-STAG avec les ACG, in *Actes de la 22^{e} conférence sur le Traitement Automatique des Langues Naturelles (TALN 2015)*, pp. 158–169, Association pour le Traitement Automatique des Langues, Caen, France, HAL open archive: hal-01145994.

Laurence DANLOS, Aleksandre MASKHARASHVILI, and Sylvain POGODALLA (2016), Interfacing sentential and discourse TAG-based grammars, in *Proceedings of the 12th International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG+12)*, Düsseldorf, Germany, HAL open archive: hal-01328697. ACL anthology: W16-3303.

Donald DAVIDSON (2001), *Essays on actions and events*, volume 1 of *Philosophical essays of Donald Davidson*, Clarendon Press.

Philippe DE GROOTE (2001), Towards Abstract Categorial Grammars, in *Proceedings of 39th Annual Meeting of the Association for Computational Linguistics (ACL 2001)*, pp. 148–155, ACL anthology: P01-1033.

Philippe DE GROOTE (2002), Tree-Adjoining Grammars as Abstract Categorial Grammars, in Frank (2002), pp. 145–150, ACL anthology: W02-2220.

Philippe DE GROOTE (2006), Towards a Montagovian account of dynamics, in Masayuki GIBSON and Jonathan HOWELL, editors, *Proceedings of the 16th Semantics and Linguistic Theory Conference (SALT 16)*, 10.3765/salt.v16i0.2952.

Philippe DE GROOTE (2015), Abstract Categorial parsing as linear logic programming, in *Proceedings of the 14th Meeting on the Mathematics of Language (MoL 2015)*, pp. 15–25, Association for Computational Linguistics, Chicago, United States, HAL open archive: hal-01188632. ACL anthology: W15-2302.

Philippe DE GROOTE and Makoto KANAZAWA (2013), A note on intensionalization, *Journal of Logic, Language and Information*, 22(2):173–194, 10.1007/s10849-013-9173-9, HAL open archive: hal-00909207.

Philippe DE GROOTE and Sarah MAAREK (2007), Type-theoretic extensions of Abstract Categorial Grammars, in *New directions in type-theoretic grammars: proceedings of the workshop*, pp. 18–30, `http://let.uvt.nl/general/people/rmuskens/ndttg/ndttg2007.pdf`.

Philippe DE GROOTE and Sylvain POGODALLA (2004), On the expressive power of Abstract Categorial Grammars: representing context-free formalisms,

Journal of Logic, Language and Information, 13(4):421–438, 10.1007/s10849-004-2114-x, HAL open archive: inria-00112956.

Philippe DE GROOTE, Ryo YOSHINAKA, and Sarah MAAREK (2007), On two extensions of Abstract Categorial Grammars, in Nachum DERSHOWITZ and Andrei VORONKOV, editors, *Proceedings of the 14th international conference on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR 2007)*, volume 4790 of *Lecture notes in computer science*, pp. 273–287, Springer, 10.1007/978-3-540-75560-9_21.

Éric Villemonte DE LA CLERGERIE (2005), From metagrammars to factorized TAG/TIG parsers, in *Proceedings of the Ninth International Workshop on Parsing Technology*, pp. 190–191, Association for Computational Linguistics, Vancouver, BC, Canada, ACL anthology: W05-1522.

Marc DYMETMAN (1994), Inherently reversible grammars, in Tomek STRZALKOWSKI, editor, *Reversible grammars in natural language processing*, chapter 2, pp. 33–57, Kluwer Academic Publishers.

Markus EGG, Alexander KOLLER, and Joachim NIEHREN (2001), The constraint language for lambda structures, *Journal of Logic, Language and Information*, 10(4):457–485, 10.1023/A:1017964622902.

Katherine FORBES, Eleni MILTSAKAKI, Rashmi PRASAD, Anoop SARKAR, Aravind K. JOSHI, and Bonnie Lynn WEBBER (2003), D-LTAG system: discourse parsing with a Lexicalized Tree-Adjoining Grammar, *Journal of Logic, Language and Information*, 12(3):261–279, 10.1023/A:1024137719751.

Katherine FORBES-RILEY, Bonnie Lynn WEBBER, and Aravind K. JOSHI (2006), Computing discourse semantics: the predicate-argument semantics of discourse connectives in D-LTAG, *Journal of Semantics*, 23(1):55–106, 10.1093/jos/ffh032.

Robert FRANK, editor (2002), *Proceedings of the Sixth International Workshop on Tree Adjoining Grammars and Related Frameworks (TAG+6)*, Università di Venezia, ACL anthology: W02-22.

Claire GARDENT (2008), Integrating a unification-based semantics in a large scale Lexicalised Tree Adjoining Grammar for French, in *Proceedings of the 22nd International Conference on Computational Linguistics (COLING 2008)*, pp. 249–256, ACL anthology: C08-1032.

Claire GARDENT and Laura KALLMEYER (2003), Semantic construction in Feature-Based TAG, in *Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2003)*, pp. 123–130, ACL anthology: E03-1030.

Claire GARDENT and Yannick PARMENTIER (2005), Large scale semantic construction for Tree Adjoining Grammars, in Blache *et al.* (2005), pp. 131–146, 10.1007/11422532_9.

Jean-Yves GIRARD (1987), Linear logic, *Theoretical Computer Science*, 50(1):1–102, 10.1016/0304-3975(87)90045-4.

Jeroen GROENENDIJK and Martin STOKHOF (1991), Dynamic predicate logic, *Linguistics and Philosophy*, 14(1):39–100, 10.1007/BF00628304.

Chung-hye HAN and Giorgio SATTA, editors (2012), *Proceedings of the 11th International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG+11)*, INRIA Paris Rocquencourt & Université Paris Diderot, Paris, France, ACL anthology: W12-4600.

Mathieu HUOT (2017), *Conservative extensions of Montague semantics*, Master's thesis, ENS Cachan, Université Paris-Saclay.

Ray JACKENDOFF (2002), *Foundations of language: brain, meaning, grammar, evolution*, Oxford University Press.

Aravind K. JOSHI (1985), Tree-adjoining grammars: how much context sensitivity is required to provide reasonable structural descriptions?, in David R. DOWTY, Lauri KARTTUNEN, and Arnold M. ZWICKY, editors, *Natural language parsing*, pp. 206–250, Cambridge University Press.

Aravind K. JOSHI (1994), Preface, *Computational Intelligence*, 10(4):VII–XV, 10.1111/j.1467-8640.1994.tb00002.x.

Aravind K. JOSHI, Laura KALLMEYER, and Maribel ROMERO (2003), Flexible composition in LTAG: quantifier scope and inverse linking, in Harry BUNT, Ielka VAN DER SLUIS, and Roser MORANTE, editors, *Proceedings of the Fifth International Workshop on Computational Semantics (IWCS-5)*.

Aravind K. JOSHI, Leon S. LEVY, and Masako TAKAHASHI (1975), Tree adjunct grammars, *Journal of Computer and System Sciences*, 10(1):136–163, 10.1016/S0022-0000(75)80019-5.

Aravind K. JOSHI and Yves SCHABES (1997), Tree-adjoining grammars, in Grzegorz ROZENBERG and Arto K. SALOMAA, editors, *Handbook of formal languages*, volume 3, chapter 2, Springer.

Sylvain KAHANE, Marie-Hélène CANDITO, and Yannick DE KERCADIO (2000), An alternative description of extractions in TAG, in *Proceedings of the 5th International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG+5)*, Université Paris 7, Jussieu, Paris, France, ACL anthology: W00-2016.

Laura KALLMEYER (2002), Using an enriched TAG derivation structure as basis for semantics, in Frank (2002), pp. 127–136, ACL anthology: W02-2218.

Laura KALLMEYER (2010), *Parsing beyond context-free grammars*, Cognitive Technologies, Springer, 10.1007/978-3-642-14846-0.

Laura KALLMEYER and Aravind K. JOSHI (2003), Factoring predicate argument and scope semantics: underspecified semantics with LTAG, *Research on Language and Computation*, 1(1–2):3–58, 10.1023/A:1024564228892.

Laura KALLMEYER and Marco KUHLMANN (2012), A formal model for plausible dependencies in Lexicalized Tree Adjoining Grammar, in Han and Satta (2012), pp. 108–116, ACL anthology: W12-4613.

Laura KALLMEYER and Maribel ROMERO (2004), LTAG semantics with semantic unification, in Rambow and Stone (2004), pp. 155–162, ACL anthology: W04-3321.

Laura KALLMEYER and Maribel ROMERO (2008), Scope and situation binding for LTAG, *Research on Language and Computation*, 6(1):3–52, 10.1007/s11168-008-9046-6.

Hans KAMP and Uwe REYLE (1993), *From discourse to logic*, Kluwer Academic Publishers.

Makoto KANAZAWA (2007), Parsing and generation as Datalog queries, in *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics (ACL 2007)*, pp. 176–183, Association for Computational Linguistics, Prague, Czech Republic, ACL anthology: P07-1023.

Makoto KANAZAWA (2008a), Prefix-correct Earley parsing of mildly context-sensitive languages, invited talk at the 15th Workshop on Logic, Language, Information and Computation (WoLLIC 2008), Edinburgh, Scotland.

Makoto KANAZAWA (2008b), A prefix-correct Earley recognizer for multiple context-free grammars, in Claire GARDENT and Anoop SARKAR, editors, *Proceedings of the Ninth International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG+9)*, pp. 49–56, University of Tübingen, Tübingen, Germany, ACL anthology: W08-2307.

Makoto KANAZAWA (2009), Second-order Abstract Categorial Grammars as Hyperedge Replacement Grammars, *Journal of Logic, Language and Information*, 19(2):137–161, 10.1007/s10849-009-9109-6.

Makoto KANAZAWA (2017), Parsing and generation as Datalog query evaluation, *IfCoLog Journal of Logics and their Applications*, 4(4):1103–1211, `http://www.collegepublications.co.uk/downloads/ifcolog00013.pdf#page=307`.

Makoto KANAZAWA and Sylvain SALVATI (2007), Generating control languages with Abstract Categorial Grammars, in Gerald PENN, editor, *Proceedings of the 12th conference on Formal Grammar (FG 2007)*, CSLI Publications, `http://research.nii.ac.jp/~kanazawa/publications/control.pdf`.

Robert KASPER, Bernd KIEFER, Klaus NETTER, and K. VIJAY-SHANKER (1995), Compilation of HPSG to TAG, in *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL 1995)*, pp. 92–99, Association for Computational Linguistics, Cambridge, MA, USA, 10.3115/981658.981671, ACL anthology: P95-1013.

Gregory M. KOBELE (2007), Parsing elliptical structure, `https://home.uni-leipzig.de/gkobele/files/unpub/Kobele07ParsingEllipsis.pdf`, unpublished ms.

Gregory M. KOBELE (2012), Idioms and extended transducers, in Han and Satta (2012), pp. 153–161, ACL anthology: W12-4618.

Gregory M. KOBELE and Jens MICHAELIS (2012), CoTAGs and ACGs, in Denis BÉCHET and Alexander DIKOVSKY, editors, *Proceedings of the 7th international conference on Logical Aspects of Computational Linguistics (LACL 2012)*, volume 7351 of *Lecture notes in computer science*, pp. 119–134, Springer, 10.1007/978-3-642-31262-5_8.

Alexander KOLLER and Marco KUHLMANN (2011), A generalized view on parsing and translation, in *Proceedings of the 12th International Conference on Parsing Technologies*, pp. 2–13, Association for Computational Linguistics, Dublin, Ireland, ACL anthology: W11-2902.

Alexander KOLLER and Marco KUHLMANN (2012), Decomposing TAG parsing algorithms using simple algebraizations, in Han and Satta (2012), pp. 135–143, ACL anthology: W12-4616.

Marco KUHLMANN and Mathias MÖHL (2007), Mildly context-sensitive dependency languages, in *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics (ACL 2007)*, pp. 160–167, Association for Computational Linguistics, Prague, Czech Republic, ACL anthology: P07-1021.

Joachim LAMBEK (1958), The mathematics of sentence structure, *American Mathematical Monthly*, 65(3):154–170, 10.2307/2310058.

Scott MARTIN and Carl POLLARD (2014), A dynamic categorial grammar, in Glyn MORRILL, Reinhard MUSKENS, Rainer OSSWALD, and Frank RICHTER, editors, *Proceedings of the 19th conference on Formal Grammar (FG 2014)*, volume 8612 of *Lecture notes in computer science*, pp. 138–154, Springer, 10.1007/978-3-662-44121-3_9.

Aleksandre MASKHARASHVILI and Sylvain POGODALLA (2013), Constituency and dependency relationship from a Tree Adjoining Grammar and Abstract Categorial Grammar perspective, in *Proceedings of the 6th International Joint Conference on Natural Language Processing (IJCNLP 2013)*, pp. 1257–1263, The Asian Federation of Natural Language Processing, Nagoya, Japan, HAL open archive: hal-00868363. ACL anthology: I13-1179.

Richard MONTAGUE (1973), The proper treatment of quantification in ordinary English, in Jaakko HINTIKKA, Julius MORAVCSIK, and Patrick SUPPES, editors, *Approaches to natural language: proceedings of the 1970 Stanford workshop on grammar and semantics*, pp. 221–242, D. Reidel Publishing Co.

Rebecca Nancy NESSON (2009), *Synchronous and Multicomponent Tree-Adjoining Grammars: complexity, algorithms, and applications*, Ph.D. thesis, Harvard

University, https://pdfs.semanticscholar.org/754b/6acaf2660748967d1937a25222538207aabc.pdf.

Rebecca Nancy NESSON and Stuart M. SHIEBER (2006), Simpler TAG semantics through synchronization, in *Proceedings of the 11th conference on Formal Grammar (FG 2006)*, CSLI Publications, http://cslipublications.stanford.edu/FG/2006/nesson.pdf.

Sylvain POGODALLA (2004a), Computing semantic representation: towards ACG abstract terms as derivation trees, in Rambow and Stone (2004), pp. 64–71, HAL open archive: inria-00107768. ACL anthology: W04-3309.

Sylvain POGODALLA (2004b), Using and extending the ACG technology: endowing categorial grammars with an underspecified semantic representation, in *Proceedings of the Categorial Grammars conference*, pp. 197–209, Montpellier, France, HAL open archive: inria-00108117.

Sylvain POGODALLA (2007a), Ambiguïté de portée et approche fonctionnelle des TAG, in *Actes de la* 14^{e} *conférence sur le Traitement Automatique des Langues Naturelles (TALN 2007)*, pp. 325–334, Association pour le Traitement Automatique des Langues, Toulouse, France, HAL open archive: inria-00141913.

Sylvain POGODALLA (2007b), Generalizing a proof-theoretic account of scope ambiguity, in *7th International Workshop on Computational Semantics (IWCS-7)*, Tilburg, Netherlands, HAL open archive: inria-00112898.

Sylvain POGODALLA (2009), Advances in Abstract Categorial Grammars: language theory and linguistic modeling. ESSLLI 2009 Lecture Notes, Part II, HAL open archive: hal-00749297.

Florent POMPIGNE (2013), *Modélisation logique de la langue et Grammaires Catégorielles Abstraites*, Ph.D. thesis, Université de Lorraine, HAL open archive: tel-00921040.

Owen RAMBOW and Matthew STONE, editors (2004), *Proceedings of the 7th International Workshop on Tree Adjoining Grammar and Related Formalisms (TAG+7)*, Simon Fraser University, Vancouver, BC, Canada, ACL anthology: W04-3300.

Owen RAMBOW, K. VIJAY-SHANKER, and David WEIR (1995), D-Tree Grammars, in *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL 1995)*, pp. 151–158, Association for Computational Linguistics, Cambridge, MA, USA, 10.3115/981658.981679, ACL anthology: P95-1021.

Owen RAMBOW, K. VIJAY-SHANKER, and David WEIR (2001), D-Tree Substitution Grammars, *Computational Linguistics*, 27(1):87–121, 10.1162/089120101300346813, ACL anthology: J01-1004.

Jim ROGERS (1999), Generalized Tree-Adjoining Grammar, in *Proceedings of the Sixth Meeting on Mathematics of Language (MoL 6)*, Orlando, FL, USA, http://www.cs.earlham.edu/~jrogers/mol6.pdf.

Sylvain SALVATI (2005), *Problèmes de filtrage et problèmes d'analyse pour les grammaires catégorielles abstraites*, Ph.D. thesis, Institut National Polytechnique de Lorraine.

Sylvain SALVATI (2006), Encoding second order string ACG with deterministic tree walking transducers, in Shuly WINTNER, editor, *Proceedings of the 11th conference on Formal Grammar (FG 2006)*, pp. 143–156, CSLI Publications, http://cslipublications.stanford.edu/FG/2006/salvati.pdf.

Sylvain SALVATI (2007), On the complexity of Abstract Categorial Grammars, in *Proceedings of the Tenth Meeting on Mathematics of Language (MoL 10)*, http://wwwhomes.uni-bielefeld.de/mkracht/mol10/abstracts/acg_complexity.pdf.

Sylvain SALVATI (2010), On the membership problem for non-linear Abstract Categorial Grammars, *Journal of Logic, Language and Information*, 19(2):163–183, 10.1007/s10849-009-9110-0.

Yves SCHABES and Stuart M. SHIEBER (1994), An alternative conception of tree-adjoining derivation, *Computational Linguistics*, 20(1):91–124, ACL anthology: J94-1004.

Stuart M. SHIEBER (1988), A uniform architecture for parsing and generation, in Dénes VARGHA, editor, *Proceedings of the 12th International Conference on Computational Linguistics (COLING 1988)*, volume 2, pp. 614–619, Budapest, Hungary, ACL anthology: C88-2128.

Stuart M. SHIEBER (1993), The problem of logical-form equivalence, *Computational Linguistics*, 19(1):179–190, ACL anthology: J93-1008.

Stuart M. SHIEBER (1994), Restricting the weak-generative capacity of Synchronous Tree-Adjoining Grammars, *Computational Intelligence*, 10(4):371–385, 10.1111/j.1467-8640.1994.tb00003.x.

Stuart M. SHIEBER (2004), Synchronous grammars as tree transducers, in Rambow and Stone (2004), pp. 88–95, ACL anthology: W04-3312.

Stuart M. SHIEBER (2006), Unifying Synchronous Tree-Adjoining Grammars and tree transducers via bimorphisms, in *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2006)*, pp. 377–384, Trento, Italy, ACL anthology: E06-1048.

Stuart M. SHIEBER (2014), Bimorphisms and synchronous grammars, *Journal of Language Modelling*, 2(1):51–104, 10.15398/jlm.v2i1.84.

Stuart M. SHIEBER and Yves SCHABES (1990), Synchronous Tree-Adjoining Grammars, in *Proceedings of the 13th International Conference on Computational Linguistics (COLING 1990)*, volume 3, pp. 253–258, Helsinki, Finland, 10.3115/991146.991191.

Stuart M. SHIEBER, Gertjan VAN NOORD, Robert C. MOORE, and Fernando C. N. PEREIRA (1989), A semantic-head-driven generation algorithm for unification-based formalisms, in *Proceedings of the 27th Annual Meeting of the Association for Computational Linguistics (ACL 1989)*, pp. 7–17, Association for Computational Linguistics, Vancouver, BC, Canada, 10.3115/981623.981625, ACL anthology: P89-1002.

Mark STEEDMAN (2001), *The syntactic process*, The MIT Press.

Mark STEEDMAN and Jason BALDRIDGE (2011), Combinatory Categorial Grammar, in Robert BORSLEY and Kersti BÖRJARS, editors, *Non-transformational syntax: formal and explicit models of grammar*, chapter 5, Wiley-Blackwell.

Johan VAN BENTHEM (1986), *Essays in logical semantics*, volume 39 of *Studies in linguistics and philosophy*, Springer, 10.1007/978-94-009-4540-1.

K. VIJAY-SHANKER (1987), *A study of Tree Adjoining Grammars*, Ph.D. thesis, University of Pennsylvania.

K. VIJAY-SHANKER (1992), Using descriptions of trees in a Tree Adjoining Grammar, *Computational Linguistics*, 18(4):481–518.

K. VIJAY-SHANKER and Aravind K. JOSHI (1985), Some computational properties of Tree Adjoining Grammars, in *Proceedings of the 23rd Annual Meeting of the Association for Computational Linguistics (ACL 1985)*, pp. 82–93, Association for Computational Linguistics, Chicago, IL, USA, 10.3115/981210.981221, ACL anthology: P85-1011.

K. VIJAY-SHANKER and Aravind K. JOSHI (1988), Feature structures based Tree Adjoining Grammars, in Dénes VARGHA, editor, *Proceedings of the 12th International Conference on Computational Linguistics (COLING 1988)*, volume 2, pp. 714–718, ACL anthology: C88-2147.

K. VIJAY-SHANKER and Aravind K. JOSHI (1991), Unification-based Tree Adjoining Grammars, Technical Report MS-CIS-91-25, University of Pennsylvania Department of Computer and Information Science (CIS), `http://repository.upenn.edu/cis_reports/762`, paper 762.

Bonnie Lynn WEBBER (2004), D-LTAG: extending lexicalized TAG to discourse, *Cognitive Science*, 28(5):751–779, 10.1207/s15516709cog2805_6.

Bonnie Lynn WEBBER and Aravind K. JOSHI (1998), Anchoring a Lexicalized Tree-Adjoining Grammar for discourse, in Manfred STEDE, Leo WANNER, and Eduard HOVY, editors, *Proceedings of the ACL/COLING workshop on discourse relations and discourse markers*, ACL anthology: W98-0315.

David J. WEIR (1988), *Characterizing mildly context-sensitive grammar formalisms*, Ph.D. thesis, University of Pennsylvania.

Sean Michael WILLIFORD (1993), *Application of Synchronous Tree-Adjoining Grammar to quantifier scoping in English*, Bachelor's thesis, Harvard College, `http://nrs.harvard.edu/urn-3:HUL.InstRepos:10951941`.

Fei XIA (2001), *Automatic grammar generation from two different perspectives*, Ph.D. thesis, University of Pennsylvania, `ftp://ftp.cis.upenn.edu/pub/fxia/thesis/thesis.pdf`.

Fei XIA, Chung-hye HAN, Martha PALMER, and Aravind K. JOSHI (2000), Comparing lexicalized treebank grammars extracted from Chinese, Korean, and English corpora, in *Second Chinese Language Processing Workshop*, pp. 52–59, Association for Computational Linguistics, Hong Kong, China, 10.3115/1117769.1117778, ACL anthology: W00-1208.

Fei XIA, Martha PALMER, and K. VIJAY-SHANKER (2005), Automatically generating Tree Adjoining Grammars from abstract specifications, *Computational Intelligence*, 21(3):246–285, 10.1111/j.1467-8640.2005.00273.x.

XTAG RESEARCH GROUP (2001), A Lexicalized Tree Adjoining Grammar for English, Technical Report IRCS-01-03, IRCS, University of Pennsylvania, `ftp://ftp.cis.upenn.edu/pub/xtag/release-2.24.2001/tech-report.pdf`.

Ryo YOSHINAKA (2006), Linearization of affine Abstract Categorial Grammars, in *Proceedings of the 11th conference on Formal Grammar (FG 2006)*, `https://web.stanford.edu/group/cslipublications/cslipublications/FG/2006/yoshinaka.pdf`.

Ryo YOSHINAKA and Makoto KANAZAWA (2005), The complexity and generative capacity of lexicalized Abstract Categorial Grammars, in Blache *et al.* (2005), pp. 330–348, 10.1007/11422532_22.

Erotetic Reasoning Corpus. A data set for research on natural question processing

Paweł Łupkowski[1,2,*], *Mariusz Urbański*[1,2], *Andrzej Wiśniewski*[1], *Wojciech Błądek*[2], *Agata Juska*[2], *Anna Kostrzewa*[2], *Dominika Pankow*[2], *Katarzyna Paluszkiewicz*[1,2], *Oliwia Ignaszak*[2], *Joanna Urbańska*[1], *Natalia Żyluk*[1,2], *Andrzej Gajda*[1,2], *and Bartosz Marciniak*
[1] Institute of Psychology, Adam Mickiewicz University, Poznań
[2] Reasoning Research Group, Adam Mickiewicz University, Poznań

ABSTRACT

The aim of this paper is to present the Erotetic Reasoning Corpus (ERC) which constitutes a data set for research on natural question processing. We describe the theoretical background, linguistic data and tags used for the annotation process. We also discuss the potential areas in which the ERC can be exploited.

Keywords: questions, logic of questions, question processing, erotetic reasoning, corpus annotation

1 INTRODUCTION

The aim of this paper is to present a data set for research on natural question processing named the Erotetic Reasoning Corpus (hereafter ERC).[1] In discourse, interlocutors must deal with question processing in instances when questions are not followed by answers but by new questions or strategies of reducing said questions into auxiliary ques-

[*] P. Łupkowski, M. Urbański and A. Wiśniewski designed the ERC and data-collection process, super-annotated the corpus and wrote the paper. W. Błądek, A. Juska, A. Kostrzewa and D. Pankow annotated the ERC. K. Paluszkiewicz, O. Ignaszak, N. Żyluk and J. Urbańska contributed to the linguistic data collection. A. Gajda and B. Marciniak implemented parts of the ERC interface.

[1] The term 'erotetic' stems from Greek 'erotema' meaning 'question'. The logic of question is sometimes called erotetic logic. For an overview of logically oriented approaches to questions and questioning see, e.g., Harrah (2002), or Wiśniewski (2015).

tions[2]. Usually, such a situation takes place when an agent wants to solve a certain problem (expressed in the form of an initial question) but is not able to reach the solution using his/her own information resources. Thus, new data, collected via questioning are necessary. This phenomenon is studied within such theoretical frameworks as Inferential Erotetic Logic (see Wiśniewski 1995, 2013, Łupkowski 2016), inquisitive semantics (see Groenendijk and Roelofsen 2011), or KoS (see Ginzburg 2012, Łupkowski and Ginzburg 2013, 2016). Natural question processing also constitutes an interesting subject for empirical research. In order to facilitate research concerning question processing in natural language dialogues, we have decided to construct the ERC. The corpus consists of the linguistic data collected in our previous studies on the question processing phenomenon. The data are annotated with a tagset, making them easy to browse for reasoning structure, pragmatic features used, and the presence of normative erotetic concepts (see Section 2).

The paper is structured as follows. We start by presenting the basic concepts of natural question processing as modelled in Inferential Erotetic Logic. We use these concepts as a normative yardstick for our design choices for the ERC tag set. Afterwards, we describe the architecture of the ERC and the linguistic data used for the corpus. Then, we introduce the tagging schema designed and used for the ERC, describe the tagging process, and discuss selected issues concerning annotation reliability. We conclude with a summary of the current stage of the project and discussion of potential future developments and applications of the ERC.

2 MODELLING QUESTION PROCESSING IN INFERENTIAL EROTETIC LOGIC

In this section, we present the underlying erotetic logic concepts used for the ERC. Our logical framework of choice is that of the Inferential Erotetic Logic (IEL; see Wiśniewski 1995, 2013). This logic focuses on inferences whose premises and/or conclusions are questions (erotetic inferences). This choice was motivated by several factors. Here, we

[2]For more details see https://intquestpro.wordpress.com/.

only mention some of them – for a detailed discussion see Urbański *et al.* (2016a). Firstly, IEL is flexible: it is not tied up to any specific logic of declaratives. Secondly, the formal representation of questions employed in IEL is friendly to the user. In general, these representations fall under the schema $?\Theta$, where Θ is an object-language expression that is equiform to a metalanguage expression which denotes the set of direct answers to a question. For example, $?\{A_1, ..., A_n\}$ represents a question whose set of direct answers is the finite set of declarative formulas: $\{A_1, \ldots, A_n\}$.[3] Yet, questions are object-language expressions of a strictly defined form and have meanings on their own; the approach is still a non-reductionistic one (see Belnap 1986; Wiśniewski 1995, pp. 37–42). On the other hand, this approach inherits the advantages of the so-called *set-of-answers methodology* (Harrah 2002; see Peliš 2016, for a comprehensive introduction, and Wiśniewski 2013, pp. 16–17 for a discussion of the semi-reductionistic approach sketched above), whose idea stems from Hamblin's (1958, p. 162) postulate: "Knowing what counts as an answer is equivalent to knowing the question." Thirdly, IEL offers some straightforward tools for modelling erotetic inferences. What is especially important from our perspective is that IEL proposes some criteria for the validity of erotetic inferences. In the case of erotetic inferences which lead from an initial question and a (possibly empty) set of declarative premises to a question, the following criteria of validity are proposed:

1. *transmission of truth/soundness into soundness*: if the initial question is sound (i.e., there exists a true direct answer to this question) and all the declarative premises, if there are any, are true, then the question which is the conclusion must be sound;
2. *cognitive usefulness*: each direct answer to a question which is the conclusion is useful in answering the initial question by narrowing down the "space of possibilities" offered by the initial question (more precisely: for each direct answer B to the question which is the conclusion there exists a non-empty proper subset Y of the set of direct answers to the initial question such that Y must contain a true direct answer to the initial question if B is true and the declarative premises, if there are any, are true).

[3] Thus $A_1, ..., A_n$ are pairwise syntactically distinct formulas.

Valid erotetic inferences (of the above kind) can be defined as those in which *erotetic implication* (e-implication for short) holds between the initial question, the declarative premises, and the question which is the conclusion. As a matter of fact, the formal definition of e-implication offers precise explications for conditions of transmission of truth/soundness into soundness and of cognitive usefulness (Definition 1; see Wiśniewski 2013, p. 68). For the sake of simplicity, we consider here only questions with finite sets of direct answers, and assume that the underlying logic of declaratives is Classical Logic. Given this, erotetic implication can be defined as follows.

Definition 1 (Erotetic implication)**.** *A question Q e-implies a question Q_1 on the basis of a set X of declaratives ($Im(Q, X, Q_1)$) iff:*

1. *for each direct answer A to the question Q: $X \cup \{A\}$ entails a disjunction of all the direct answers to the question Q_1, and*
2. *for each direct answer B to the question Q_1 there exists a non-empty proper subset Y of the set of direct answers to the question Q such that $X \cup \{B\}$ entails a disjunction of all the elements of Y.*

It is easily seen that clauses (1) and (2) of Definition 1 mirror the criteria of validity discussed above.

Applying erotetic implication for modelling certain real-life linguistic phenomena resulted in identifying two other versions of this kind of relation, weaker than the one just defined (which we shall further on call the canonical erotetic implication). These are the weak erotetic implication (Urbański *et al.* 2016a) and the falsificationist erotetic implication (Grobler 2012; Wiśniewski 2013), both of which modify the second condition of the original definition.

Definition 2 (Weak erotetic implication)**.** *A question Q weakly e-implies a question Q_1 on the basis of a set X of declaratives ($Im_w(Q, X, Q_1)$) iff:*

1. *for each direct answer A to the question Q: $X \cup \{A\}$ entails a disjunction of all the direct answers to the implied question Q_1, and*
2. *for some direct answer B to the question Q_1 there exists a non-empty proper subset Y of the set of direct answers to the question Q such that $X \cup \{B\}$ entails a disjunction of all the elements of Y.*

Definition 3 (Falsificationist erotetic implication)**.** *A question Q f-implies a question Q_1 on the basis of a set X of declaratives ($Im_f(Q, X, Q_1)$) iff:*

1. *for each direct answer A to the question Q: $X \cup \{A\}$ entails a disjunction of all the direct answers to the question Q_1, and*
2. *for some direct answer B to the question Q_1, $X \cup \{B\}$ eliminates at least one direct answer to Q.*

The concept of elimination used in Definition 3 is construed as follows: a formula A *eliminates* a formula B just in case B must be false if A is true, given the underlying semantics (for a precise definition see Wiśniewski 2013, p. 34).

The properties described in the second clauses of definitions 1, 2, and 3 will be referred to below as 'usefulness', 'w-usefulness', and 'f-usefulness', respectively.

Table 1 presents examples of erotetic implication of the three presented types.

Q, X, Q_1	e-implication
$?\{p, q \vee r\}, \emptyset, ?\{p, q, r\}$	Im
$?p, p \leftrightarrow q, ?q$	Im
$?\{\neg p, r, s\}, \emptyset, ?\{p, q, \neg q\}$	Im_f
$?\{p, q, v\}, s \rightarrow p, ?\{s, \neg s\}$	Im_w
$?\{\neg p, r, s\}, \neg p \vee r \vee s, ?\{p, q, \neg q\}$	Im_w, Im_f
$?\{p, q, w\}, p \vee q \rightarrow r, p \vee q \vee w, ?\{r, \neg r\}$	Im_w, Im_f
$?\{p, q, v\}, p \vee q, r \leftrightarrow q, ?\{r, \neg r\}$	Im, Im_w, Im_f

Table 1:
Examples of canonical (Im), weak (Im_w) and falsificationist (Im_f) erotetic implication

Notions introduced in this section will be reflected by the tagset used to annotate the ERC, described in detail in Section 4 of the present paper.

Using e-implication as a tool allows for modelling many aspects of natural question processing, i.e. a situation in which an initial question is internally processed by an agent, and where the outcome is either a new question concerning the subject matter or a strategy of reducing the initial question into auxiliary questions. In both cases, e-implication allows for the description and assessment of the inferences which lead from questions to questions.

The basic areas of applicability of the analysis of the described phenomena include: the search for information in distributed resources, question answering (in particular, cooperative answering), problem solving (in particular, problem solving by interrogation), proof theory and automated deduction (proof search, complexity issues).

3 LINGUISTIC DATA

The linguistic data used for the ERC were gathered for research on question processing. The outcomes of three research projects are employed here. These are: the Erotetic Reasoning Test, QuestGen and Mind Maze.

The Erotetic Reasoning Test (in Polish: Test Rozumowań Erotetycznych, TRE) is a tool used in the research described in detail in (Urbański *et al.* 2016a). The test contains 3 items (with an imposed time limit of 30 min). Each item consists of a detective-like story in which the initial problem and evidence gained are indicated. The task is to pick a question (one out of four), each answer to which will lead to some solution to the initial problem. The subjects are asked to justify their choices.

Let us present here an exemplary tasks from TRE (translated into English). The task is entitled "The Bomb":

> In the capital of a certain country someone planted a bomb in the palace of the king. The best royal engineer, who arrived immediately, established the following facts:
> 1. There are three wires in the bomb: green, red and orange;
> 2. To disarm the bomb either the green or the red wire must be cut. Cutting the wrong wire will cause an explosion;
> 3. If the bomb has been planted by Steve, cutting the green wire will disarm it;
> 4. If the bomb has been planted by John, cutting the red wire will disarm it. Moreover, no one but John would have used the red wire;
> 5. If the bomb has not been planted on an even day of the month, the culprit is Steve;
> 6. The bomb has been planted either by Steve, or by John, or by someone else.

> Each of the following questions below can be answered either 'yes' or 'no'. Mark the question to which the answer (regardless of it being 'yes' or 'no') will allow you to establish, in the shortest time possible, which wire should be cut in order to disarm the bomb:
>
> > Was the bomb planted on an even day of the month?
> >
> > Was the bomb planted by Steve?
> >
> > Was the bomb planted by John?
> >
> > Was the bomb planted by someone else than Steve or John?
>
> Justify your choice.

TRE-entries of the ERC have a well-established structure: there is a story, a question chosen by the subject and then a justification of the choice. An exemplary justification (translated into English) provided by a subject for the "Bomb" story is presented below (see Urbański *et al.* 2016a, p. 41).

> If we'll get an affirmative answer to this question, then we'll know that the green wire needs to be cut. If a negative one, then there will be only one possibility left – the red wire, and additionally we'll know that the culprit is John.

QuestGen is an online game the aim of which is to engage players in generating a large collection of questions for a certain piece of story written in a natural language (as such it might be perceived as an example of a game with a purpose – see Von Ahn and Dabbish 2008). The idea of the game was presented in (Łupkowski 2011), while its implementation is described in (Łupkowski and Wietrzycka 2015) and (Łupkowski and Ignaszak 2017). In the game, two randomly chosen players are engaged in solving a detective puzzle. One of them plays as the Detective, the other as the Informer. The Detective's objective is to solve the presented puzzle by questioning the Informer. Each story in the game has two versions (one for the Detective and one for the Informer), containing all the additional data necessary to solve the puzzle. The Detective is allowed to use only yes/no questions and cannot ask straightforwardly for the solution. The Detective may ask as

many questions as s/he wants/needs (as long as they are simple yes-no questions). The Informer is obliged to answer the Detective's questions in accordance with the information presented in the Informer's part of the story. Each story is played within a time limit. The game is played in cooperative mode, i.e. the Detective and the Informer play together constrained by the time limit and obtain points for each puzzle solved.

As an example of the task from the QuestGen game, we present the Detective's part of a story entitled "Arsen L.":

> Imagine that you are a detective who is following the well-known international villain Arsen L. You are trying to establish if Arsen L. went to Paris, London, Kiev, or Moscow. You look through your notes and this is the information you have managed to gather so far:
> 1. Arsen L. left for Paris or London if and only if he departed in the morning;
> 2. Arsen L. left for Kiev or Moscow if and only if he departed in the evening;
> 3. If Arsen L. took a train, then he did not leave for London or Moscow;
> 4. If Arsen L. left for Paris or Kiev, then he took a train.
>
> So, where did Arsen L. go?
> Before you answer this question you may ask several auxiliary questions of the railway station employee. Remember: your time is limited. Ask only yes/no questions. It is pointless to ask the employee directly about where Arsen L. went because he does not have a clue.

Solutions gathered within the QuestGen project have a well-established structure, very much like the ones from TRE. A QG-entry of the ERC consists of the story which is followed by the main question (expressing the problem to be solved by the player). Afterwards, we observe the sequence of the Detective's questions and the Informer's answers which is ended by the proposed solution to the main question and the feedback given by the Informer. This gives us more interaction than in the TRE case. We observe short dialogues between players. An

example (translated into English) of such a dialogue for the "Bomb" story is presented below (see Łupkowski and Ignaszak 2017, p. 239):

DETECTIVE: Is it the case that Anthony has something to do with de bomb?
INFORMER: No.
DETECTIVE: So it is the case that Roger is guilty?!
INFORMER: Yes.
DETECTIVE: Orange, isn't it?
INFORMER: Yes.
DETECTIVE: Orange.

Mind Maze (in Polish "Takie życie") is a card game published by Igrology. In the game, one of the players plays the role of the game master (GM) and the other one tries to solve a puzzle presented by the game master. the GM tells a short story (inspired by true events) and the objective of the player is to figure out how the story happened by asking questions to the GM. Only yes/no questions are allowed here (with two additional admissible answers: "It is not important/relevant" and "It is not known"). *Mind Maze* was used as the core element for the semi-structured study of question processing (see Urbański and Żyluk 2016 and Urbański *et al.* 2016a). The researcher played the role of the GM and subjects were players. Game sessions were recorded and then transcribed.

To give an example (translated into English) of the types of problems to solve in the Mind Maze game, let us consider the one entitled "The Traveller":

> A man without a single visa visited eight different countries in a single day. None of the authorities of these countries tried to remove him. What was his profession and how did he manage to do this?

Solutions gathered in the described study are the most complex ones in the ERC data set. They have no clear structure as they are more or less free dialogue leading to the solution of the initial problem. The shortest conversation included in the ERC has 760 words, while

Table 2:
Characteristics of the linguistic data set of ERC

Source	Files	Words
TRE	270	81.169
QG	116	21.944
TZ	16	30.619
Sum	402	133.732

the longest one is 3.367.[4] An example *Mind Maze* interaction between the player and the game master (translated into English) is presented below:

PLAYER: Is this building a cultural one?
GM: Cultural one... in what sense it is a cultural building?
PLAYER: Related to culture, history, art? Related to culture?
GM: But, how would you define this „related"?
PLAYER: Related... it is used for cultural purposes, developement related issues, for people. To some extent educational ones?

To differentiate the aforementioned sources, we will refer to them as the ERC sub-corpora, the TRE, QG, and TZ, respectively. The whole ERC consists of 402 files (solutions). Table 2 presents a summary of the gathered data. Note that all of the data are in Polish; however, the tagset used for the annotation allows for the data to be analyzed by English-speaking researchers.

4 TAGGING

The tagging schema for the ERC consists of three layers:

1. The structural layer – representing the structure of the tasks used for the studies described in Section 3. Here, we distinguish between elements such as: instructions, justifications, different types of questions, and declaratives.
2. The inferential layer – which allows for normative elements described in Section 2 to be identified.
3. The pragmatic layer – representing various events that may occur in the dialogue, like e.g. long pauses. It also contains tags that

[4] For comparison, the longest files for the TRE and the QG have 387 and 230 words respectively.

enable the expression of certain events related to the types of tasks used (like e.g. when a forbidden question – that is, question of the form which is not allowed in a certain entry – is used).

Let us now present, and explain in detail, the tags used in the ERC. Each task in the ERC is tagged with the KORPUS tag which has two obligatory attributes:

- first one specifying the sub-corpus of ERC (namely whether the task comes from Erotetic Reasoning Test: TRE, QuestGen: QG or Mind Maze: TZ),
- second one specifying the name of the task and the number of the subject/player who solved it.

4.1 *Structural layer*

The structural layer of annotation consists of the following tags: INSTRUCTION; JUSTIFICATION; DECLARATIVE; QUESTION.

- The INSTRUCTION: the tag indicates instruction for a given task.
- The JUSTIFICATION: a justification given by a subject is indicated with this tag.
- The DECLARATIVE: tag marking declaratives.
- The QUESTION: tag for indicating questions.

The DECLARATIVE and QUESTION tags enable certain attributes to specify further details. These attributes are presented in Figure 1 and 2. Pointing out one of the attributes marked with a solid line is obligatory. The ones marked with a dashed line are non-obligatory.

The QUESTION tag is associated with the following attributes:

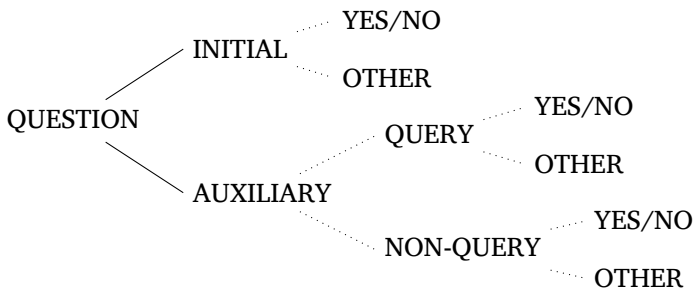


Figure 1: The QUESTION tag and its attributes

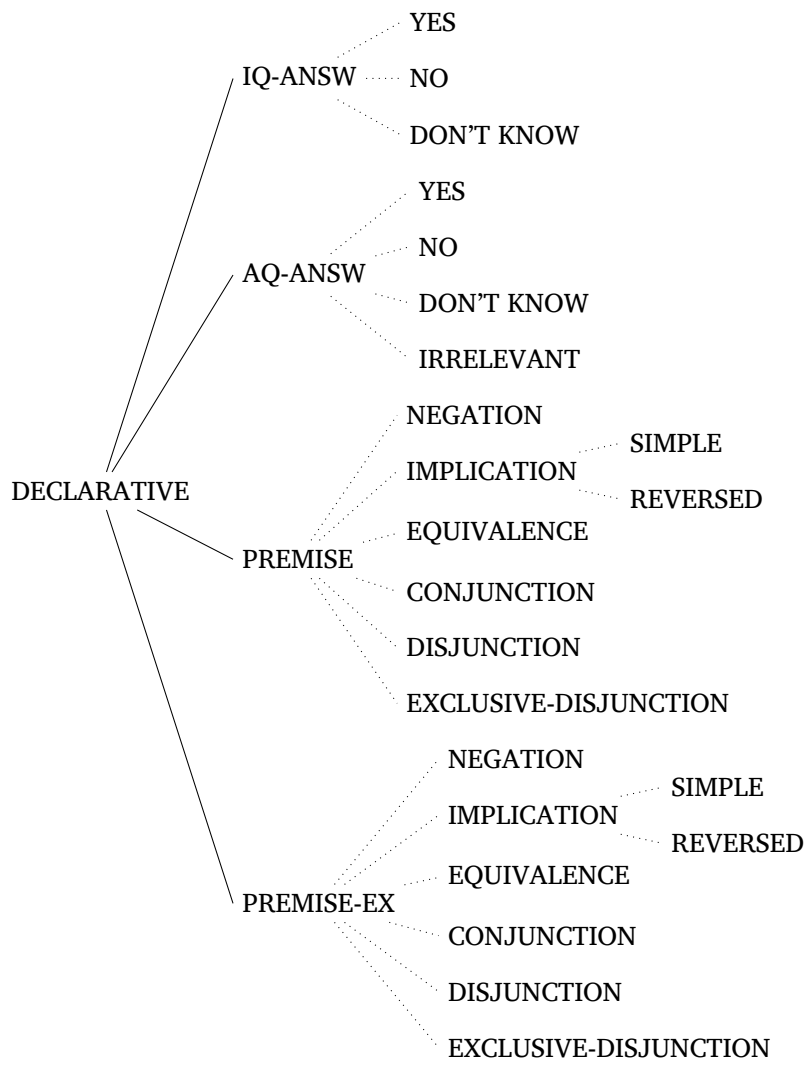


Figure 2: The DECLARATIVE tag and its attributes

1. INITIAL: points out the initial question. Additional attributes allow for specifying whether the initial question is of the yes/no or other type.
2. AUXILIARY: marks questions recognized as auxiliary ones. Attributes associated with the tag indicate whether the auxiliary question is a query and point to its type (yes/no or other type of question).

The DECLARATIVE tag is associated with the following attributes:

- IQ-ANSW: indicates an answer to the initial question. The type of answer given might be specified by: YES, NO, DON'T KNOW.
- AQ-ANSW: indicates an answer to the auxiliary question. Similarly to the IQ-ANSW case, the type of answer given might be further specified by: YES, NO, DON'T KNOW, IRRELEVANT.
- PREMISE: used for premises (declarative ones). Additional attributes may be used to specify a logical structure of the recognized premise. For the premises with the implication as the main connective a more detailed characteristics may be provided with the tags: SIMPLE or REVERSED.
- PREMISE-EX: used for a declarative premise which allows for exceptions. To exemplify such a premise, consider the following (from "The Party" task of TRE): "The King of Hearts stays till the end of only those parties at which the March Hare doesn't tell jokes (although even then the King sometimes leaves earlier)."

Additional attributes for these tags are the same as those for the PREMISE tag.

4.2 *Inferential layer*

The inferential layer consists of nine tags: SOLUTION; TRANSMISSION; USEFULNESS; W-USEFULNESS; F-USEFULNESS; E-OTHER; ENTAILMENT; D-OTHER; IMP-ERROR. This layer plays an important role in the ERC making our data set unique. The tags used here stem from the IEL's ideas and concepts presented in Section 2. This layer makes it possible to track and study how these concepts are applied and used in the context of reasonings enforced by the tasks used for our subcorpora.

SOLUTION: this tag indicates the solution given by a subject. Additional attributes allow for specifying whether the solution is correct (note that each task in the ERC has a predefined normative solution) and how this solution has been reached (i.e. whether it is in line with the assumed normative way of obtaining the solution – e.g. erotetic search scenario in the case of QG tasks). Attributes of the SOLUTION tag are presented in Figure 3.

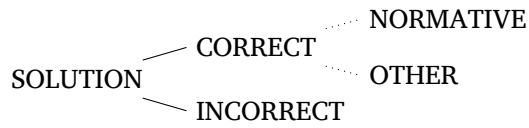


Figure 3:
The SOLUTION tag and its attributes

TRANSMISSION: this tag is used for such justifications that cover the first condition of the definition of erotetic implication, i.e. transmission of truth/soundness (including the canonical one as well as the weak one and the falsificationist one – see Definitions 2 and 3 in Section 2).

USEFULNESS: this tag is used for such justifications that cover the second condition of the definition of (canonical) erotetic implication, i.e. cognitive usefulness.

W-USEFULNESS: this tag is used for such justifications that cover the second condition of the definition of the weak erotetic implication.

F-USEFULNESS: this tag is used for such justifications that cover the second condition of the definition of the falsificationist erotetic implication

E-OTHER: marks such justifications that are not modelled by Inferential Erotetic Logic.

ENTAILMENT: this tag is used for such justifications that correctly refer to logical entailment.

D-OTHER: this tag is used for such justifications that incorrectly refer to logical entailment or to a different type of relation between declaratives.

IMP-ERROR: denotes justifications in which a subject interpreted the material implication in the incorrect way (according to Classical Logic).

4.3 *Pragmatic layer*

The pragmatic level consists of the five tags. It should be noted that certain pragmatic layer tags are used only within selected sub-corpora as described below.

Q-FORBIDDEN: allows one to point out when a forbidden question appears in the solution of tasks in the QG and TZ subcorpora. This refers to the rules provided for a given task. For example, this tag is used in the case of a QuestGen task when the Detective will ask directly

about the solution. In the Mind Maze tasks, this tag appears when a player uses a question other than that of a yes/no type.

WRONGINFO: this tag is used in the QG sub-corpus. It denotes a situation wherein the Informer provides a wrong piece of information to the Detective in the game. "Wrong", in this case, means different than the one given in the Informer's part of the story. This tag will also be used in situations in which the Detective asks a question marked as Q-FORBIDDEN and the Informer answers with something different than the desired "I don't know" answer.

KEY-INFO: is used for the TZ sub-corpus. It indicates additional information provided by the game master (the information provided is not an answer to a question in the game).

TOPIC: is also a tag used in the TZ sub-corpus for marking topics (as defined by van Kuppevelt (1995)) as they appear in a dialogue.

LONG-PAUSE: the tag is used in the QG and TZ sub-corpora for indicating long pauses in the game.

An example annotated ERC file is presented in Figure 4. The figure presents the file from the TRE sub-corpus of the ERC, the task name is "Bomb" and the file number is 31 – this is visible in the first line containing the tag <KORPUS A1 = "TRE" A2 = "Bomba31">. The structure of the file is clearly visible owing to the structural layer of the tags used. We can identify the instruction part as well as the premises and the initial question, solution, and justification provided by the subject in this case. Tags used to annotate premises provide information about their structure (visible as the A2 attribute), e.g. in the last premise, an exclusive disjunction is used. The initial question is identified by a <QUESTION> tag with the A1 attribute stating "INITIAL". The A2 attribute informs us that this is not a simple yes/no question. Let us now take a closer look at the solution, which is indicated by the following tag: <SOLUTION A1 = "CORRECT" A2 = "NORMATIVE">. Attributes of this tag inform us that the solution provided by the subject is the correct one, what is more, it is also normative. This leads us to the justification part of this file. There we find two tags: <TRANSMISSION /> and <USEFULNESS />, which provide information about the normativity of the provided correct solution – this

```
<KORPUS A1="TRE" A2="Bomba31">

<INSTRUCTION>
 Wprowadzenie: W stolicy pewnego kraju ktoś podłożył bombę w pałacu króla. Najlepszy saper
     królewski, który przybył na miejsce, ustalił sobie jedynie znanymi sposobami kilka faktów:

(a) <DECLARATIVE A1="PREMISE" A2="CONJUNCTION" A4="1">W bombie znajdują się trzy kabelki:
    zielony, czerwony i pomarańczowy.</DECLARATIVE>
(b) <DECLARATIVE A1="PREMISE" A2="EXCLUSIVE-DISJUNCTION" A4="2">Żeby unieszkodliwić bombę trzeba
    przeciąć albo zielony, albo czerwony kabelek. Przecięcie niewłaściwego kabelka spowoduje
    wybuch.</DECLARATIVE>
(c) <DECLARATIVE A1="PREMISE" A2="IMPLICATION" A3="SIMPLE" A4="3">Jeżeli bombę podłożył Stefan,
    to unieszkodliwia ją przecięcie zielonego kabelka.</DECLARATIVE>
(d) <DECLARATIVE A1="PREMISE" A2="EQUIVALENCE" A4="4">Jeżeli bombę podłożył Ignacy, to
    unieszkodliwia ją przecięcie czerwonego kabelka. Co więcej, nikt inny do tego celu nie
    wykorzystałby czerwonego kabelka.</DECLARATIVE>
(e) <DECLARATIVE A1="PREMISE" A2="IMPLICATION" A3="SIMPLE" A4="5">Jeśli bomby nie podłożono w
    dzień parzysty, to zrobił to Stefan.</DECLARATIVE>
(f) <DECLARATIVE A1="PREMISE" A2="EXCLUSIVE-DISJUNCTION" A4="6">Bombę podłożył albo Stefan, albo
    Ignacy, albo jeszcze ktoś inny.</DECLARATIVE>

Instrukcja: Na każde z poniższych pytań można uzyskać jedną z dwóch odpowiedzi:
'tak' albo 'nie'. Zaznacz symbolem 'x' tylko jedno pytanie, na które dowolna
odpowiedź (niezależnie od tego, czy będzie to 'tak' czy 'nie') pozwoli jak najszybciej
ustalić, <QUESTION A1="INITIAL" A2="OTHER">który kabelek należy przeciąć, żeby unieszkodliwić
    bombę.</QUESTION>

[ ] <QUESTION A1="AUXILIARY" A2="QUERY" A3="YES/NO" A4="1">Czy bombę podłożono w dzień parzysty?
    </QUESTION>
[ ] <QUESTION A1="AUXILIARY" A2="QUERY" A3="YES/NO" A4="2">Czy bombę podłożył Stefan?</QUESTION>
[x] <SOLUTION A1="CORRECT" A2="NORMATIVE"><QUESTION A1="AUXILIARY" A2="QUERY" A3="YES/NO"
    A4="3">Czy bombę podłożył Ignacy?</QUESTION></SOLUTION>
[ ] <QUESTION A1="AUXILIARY" A2="QUERY" A3="YES/NO" A4="4">Czy bombę podłożył ktoś inny niż
    Stefan lub Ignacy?</QUESTION>

Uzasadnij, dlaczego wybrałaś/wybrałeś to właśnie pytanie.
</INSTRUCTION>

<JUSTIFICATION>
<TRANSMISSION />
<USEFULNESS />
Bomba Stefana może mieć zielony Bomba Ignacego – kabel czerwony i zielony Bomba kogoś innego –
    kabel zielony Jeśli dowiemy się, że to Ignacy to trzeba będzie przeciąć czerwony, gdy okaże
    się, że to nie on, to w każdym innym przypadku będzie to kabel zielony niezależnie czy to
    Stefan cze ktoś inny podłożył bombę.
</JUSTIFICATION>
</KORPUS>
```

Figure 4: An exemplary annotated ERC file

warrants the conclusion that the solution provided can be modelled in terms of canonical erotetic implication (see Definition 1).

4.4 *Descriptive statistics of the annotation*

Let us now take a closer look at the descriptive statistics of the ERC annotation.

We will start with the *structural layer* of the annotation. The number of INSTRUCTION tags is the same as the number of ERC files, as each task comes with its own instruction. We have 402 INSTRUCTION

tags (270 for TRE, 116 for QG and 16 for TZ). As for the JUSTIFICATION tag, it is present only in the TRE sub-corpus and the number of these tags is equivalent to the number of TRE files in the ERC, i.e. 270. The reason for this is that each TRE solution consists of an auxiliary question indicated a subject and a justification provided for this choice (as described in Section 3). The ERC has 2.234 QUESTION tags, 1.350 in TRE sub-corpus, 375 in the QG and 527 in the TZ. Details are presented in Table 3. As for DECLARATIVE tags, there are 2.855 (TRE: 1.530, QG: 777, TZ: 548) – details are presented in Table 4.

	TRE	QG	TZ	Sum
QUESTION	1.335	357	527	2.234
INITIAL	270	116	16	402
INITIAL YES/NO	0	19	0	19
INITIAL OTHER	270	97	16	383
AUXILIARY	1.080	241	511	1.832
QUERY	1.080	238	452	1.770
QUERY YES/NO	1.080	238	442	1.760
QUERY OTHER	0	0	10	10
NON-QUERY	0	3	59	62
NON-QUERY YES/NO	0	3	13	16
NON-QUERY OTHER	0	0	46	46

Table 3: Descriptive statistics for the QUESTION tag

For the *inferential layer* we will first discuss the SOLUTION tag. The detailed numbers for this tag are presented in Table 5. The total number of occurances of the SOLUTION tag for the TZ sub-corpus is larger than the number of files. This is because the solution is divided into two parts for each file, corresponding to the dialogue structure. It should be noted that the vast majority of solutions for the ERC tasks were correct ones. (For the TZ sub-corpus NORMATIVE and OTHER attributes were not used).

For the TRE sub-corpus, additional inferential tags were also used. This is due to the structure of the solutions provided by the subjects, i.e. answers to initial questions and their corresponding justifications. There are 205 TRANSMISSION and 160 USEFULNESS tags used. For 149 cases the TRANSMISSION and USEFULNESS tags are both present, which constitutes the number of correct and normative solutions for the sub-corpus.

Table 4: Descriptive statistics for the DECLARATIVE tag

	TRE	QG	TZ	Sum
DECLARATIVE	1.530	777	548	2.855
IQ-ANSWER	0	109	11	120
YES	0	5	0	5
NO	0	10	0	10
DON'T KNOW	0	1	0	1
AQ-ANSWER	0	241	500	741
YES	0	109	191	300
NO	0	120	216	336
DON'T KNOW	0	12	21	33
IRRELEVANT	0	0	25	25
PREMISE	1.350	427	36	1.813
IMPLICATION	720	271	0	991
EQUIVALENCE	180	96	0	276
CONJUNCTION	90	0	0	90
EXCLUSIVE-DISJ	270	20	0	290

Table 5: Descriptive statistics for the SOLUTION tag

	TRE	QG	TZ	Sum
SOLUTION	268	109	17	394
CORRECT	190	91	17	298
CORRECT NORMATIVE	149	44	–	192
CORRECT OTHER	41	47	–	88
INCORRECT	78	18	0	94

Let us now discuss the *pragmatic layer* of annotation. As can be expected, there are no pragmatic tags in the ERC sub-corpus, due to the nature of the task involved. The numbers for this layer will get bigger for sub-corpora with more interaction involved. And we have 8 Q-FORBIDDEN and 29 WRONGINFO tags for the QG sub-corpus. As it was described above, the WRONGINFO tag is specific to the QG sub-corpus. The reason why this is the case for these tasks is that a randomly chosen player has to play the role of the informer in the game. S/he has to process additional information related to the puzzle and provide answers to the Detective within the specified time limit. As a result, we sometimes observe that the Informer provides wrong information. It is important to mark these utterances in the ERC, as this makes solving the puzzle harder or sometimes impossible

for the Detective. In the TZ sub-corpus, we observe more pragmatic tags, as here we are dealing with (almost) free dialogue. There are 16 Q-FORBIDDEN, 61 KEY-INFO, 438 TOPIC and 100 LONG-PAUSE tags used for these tasks. In the TZ context, especially, KEY-INFO and TOPIC are interesting as they were designed especially for this sub-corpus. TOPIC allows one to track how new topics related to the solution of a given story are introduced and resolved. As for the KEY-INFO tag, it is crucial for understanding how the solution to the initial question is reached as this tag indicates situations in which a game-master provides addition information, which facilitates the solving process.

To sum up, we observe 24 Q-FORBIDDEN, 29 WRONGINFO, 61 KEY-INFO, 438 TOPIC, and 100 LONG-PAUSE pragmatic layer tags in the ERC data. As we have mentioned, due to the nature of the tasks, these tags are present only in the QG and TZ sub-corpora of the ERC.

4.5 *Annotation and its reliability*

The tagging process was performed by 5 volunteers with solid background in erotetic logic. Each file was tagged by one annotator. What is more, each annotator tagged files only from one sub-corpus of the ERC. Thanks to this, s/he dealt with a consistent file structure and consistent subset of the tagset.

Annotation quality was ensured via a variety of measures. First of all, the structural tags layer is very intuitive and standardised for the TRE and QG sub-corpora (see description in Section 3). For these files, an experienced super-annotator (with expert knowledge in IEL) prepared and controlled the annotation schemas used. Each controversial case was discussed by the annotators.

Secondly, the output consists of XML files, thus RELAX NG XML schema was defined with the purpose of facilitating the annotation process. The schema specifies a pattern for the structure and the content of XML files and prevents incorrect use of tags by annotators. All of the ERC files were validated by the annotators themselves and afterwards by a super-annotator. The validation was performed in two steps: first general XML validity was checked and in the second step ERC XML schema were used to control the use of the ERC tagset. Structural validity was also checked within the ERC tools described below.

Thirdly, all of the ERC files were thoroughly controlled by the super-annotator. Every issue has been discussed between the annotators; and this is how final tagging was established.

In order to check the reliability of the annotation process, inter- and intra-annotator tests were performed.

For the inter-annotator test, a sample of 100 randomly chosen text units (retrieved from all three sub-corpora of ERC) was used. The units were chosen in such a way that they could be annotated with at least one ERC tag. The structure of the sample was the same as the whole ERC, i.e. 67% of units were retrieved from the TRE sub-corpus; 29% from the QG and 16% from the TZ. All of the units were supplemented with a necessary context.

The guideline for annotators contained explanations of all the ERC tags and examples of annotated text units. The control sample was annotated by two annotators (two logicians, one of whom had a solid background in the logic of question).

The reliability of the annotation was evaluated using κ (Carletta 1996), established by using the R statistical software (R Core Team 2013; version 3.3.1) with the *irr* package (Gamer *et al.* 2012). The interpretation of the kappa values is based on that of Viera and Garrett (2005).

The Fleiss κ for all three annotators was 0.8 (i.e. substantial) with 75% agreement over 100 cases. The agreements between the main annotation and others were high, as presented below:

- main and first annotator: $\kappa = 0.85$, with 86% agreement (almost perfect agreement);
- main and second annotator: $\kappa = 0.78$, with 80% agreement (substantial).

As can be expected, when it comes to a detailed analysis of the annotation, the most unproblematic cases were the ones annotated with tags from the structural and pragmatic layers of the ERC tagset. Annotation with the use of the inferential layer was more problematic. Cases where we observe disagreement between annotators concern the use of <TRANSMISSION /> and <USEFULNESS /> tags for the TRE sub-corpus samples. The reason for this may be that the use of these tags involves the interpretation of the justification provided by a subject in the light of an answer given for a particular task. As it

was explained above, we have paid special attention to this layer of annotation of the ERC. All of the tags used were checked by the super-annotator and each controversial case was discussed by the main ERC annotators.

We have also performed intra-annotator agreement rating test. For this test, another control sample of 100 examples was randomly chosen from the data (with the same structure as the sample for the inter-annotation study). In this case, two ERC annotators were employed to annotate the sample. The agreement between the main annotation and the two annotators was almost perfect – Fleiss $\kappa = 0.86$ with 82% agreement over 100 cases. The detailed results for annotators are presented below:

- main and first annotator: $\kappa = 0.87$, with 88% agreement;
- main and second annotator: $\kappa = 0.85$, with 86% agreement;
- first and second annotator: $\kappa = 0.86$, with 87% agreement.

5 ERC ON-LINE

The corpus is available via its web-site[5]. ERC is distributed under the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License.

Several tools that allow one to to work with the corpus are provided on the ERC web-site.[6] The central tool is *ERC Search & Browse Tool*. This application allows one to display and browse ERC files, both with and without tags. It also allows one to search through corpus files. Keyword and tag search options have likewise been made available to users. In order to use a certain fragment of the ERC in one's paper, presentation, or poster one may take advantage of the ERC XML/LaTeX Parser (Gajda and Łupkowski 2016). The parser transforms original XML-annotated ERC files into appropriate LaTeX files. The parser is responsible for formatting and displaying the data form the corpus – it will be especially useful for preparing papers and presentations based on the ERC data. Hence the choice of using LaTeX as the output format for our tool. Obtained files may be simply pasted into an article,

[5] See https://ercorpus.wordpress.com/

[6] See https://ercorpus.wordpress.com/tools/.

presentation, or poster.[7] The last tool provided is *ERC XML Schema*. The ERC XML Schema describes the structure of corpus XML files. It allows for quick syntactic validation of corpus files and is very useful in the annotation process.

6 SUMMARY

In this paper, we have presented the Erotetic Reasoning Corpus. So far, the ERC data have been mainly analysed in the light of the normative yardstick provided by IEL. Urbański *et al.* (2016a) present research on correlations between the level of fluid intelligence and fluencies in two kinds of deductions: simple (syllogistic reasoning) and difficult ones (erotetic reasoning). The tool used to investigate erotetic reasoning is the Erotetic Reasoning Test. The paper presents the detailed analysis of the justifications provided by subjects. Urbański *et al.* (2016b) contains analyses of solutions to Mind Maze games. Łupkowski and Ignaszak (2017) model and discuss selected solutions of QuestGen tasks with focusing on normative vs. non-normative solutions.

In our opinion, however the ERC's potential scope of use is broad and reaches far beyond studies of the normative logical concepts vs. instances of real erotetic reasoning. The ERC consists of a significant amount of natural language data (see Table 2). The potential applications may cover the following example areas of interests:

- linguistic studies of the way questions are formulated in different contexts;
- research on dialogue management (this applies in particular to the TZ sub-corpus of the TRE, which consists of long natural language dialogues);
- problem solving studies concerning strategies of handling question decomposition, especially those with imposed time limits (such as the tasks in the QG sub-corpus of the ERC);
- studies focusing on the way a question should be asked (or an initial problem/task should be formulated) in order to make the solution easier to reach.

[7] For an overview of LaTeX in academic use see e.g. (de Souza e Silva Filho and Pinheiro 2010), (Flom 2005), (Hofert and Kohm 2010), (Łupkowski 2015), (Łupkowski and Urbański 2013).

ACKNOWLEDGEMENTS

Work on the Erotetic Reasoning Corpus was supported by the National Science Centre, Poland (DEC-2013/10/E/HS1/00172 and DEC-2012/04/A/HS1/00715).

REFERENCES

Nuel Belnap (1986), Approaches to the semantics of questions in natural language: part 1, in *From models to modules*, pp. 257–284, Ablex Publishing Corp.

Jean Carletta (1996), Assessing Agreement on Classification Tasks: The Kappa Statistic, *Computational Linguistics*, 22(2):249–254.

Paulo Rogério de Souza e Silva Filho and Rian Gabriel Santos Pinheiro (2010), Design and Preparation of Effective Scientific Posters using LaTeX, *The PracTeX Journal*, 2010(2), `http://tug.org/pracjourn/2010-2/rogerio.html`.

Peter Flom (2005), LaTeX for academics and researchers who (think they) don't need it, *The PracTeX Journal*, 2005(4), `http://tug.org/pracjourn/2005-4/flom/flom.pdf`.

Andrzej Gajda and Paweł Łupkowski (2016), Using LaTeX as an element of the Erotetic Reasoning Corpus interface, in Tomasz Przechlewski, Karl Berry, and Jerzy Ludwichowski, editors, *BachoTeX 2016: Convergence*, pp. 47–52, Polish TeX Users Group GUST, Bachotek.

M. Gamer, J. Lemon, and I.F.P. Singh (2012), irr: Various Coefficients of Interrater Reliability and Agreement. R package version 0.84, http://CRAN.R-project.org/package=irr.

Jonathan Ginzburg (2012), *The Interactive Stance: Meaning for Conversation*, Oxford University Press, Oxford.

Adam Grobler (2012), Fifth part of the definition of knowledge, *Philosophica*, 86:33–50.

Jeroen Groenendijk and Floris Roelofsen (2011), Compliance, in Alain Lecomte and Samuel Tronçon, editors, *Ludics, Dialogue and Interaction*, pp. 161–173, Springer-Verlag, Berlin Heidelberg.

C. L. Hamblin (1958), Questions, *The Australasian Journal of Philosophy*, 36:159–168.

David Harrah (2002), The Logic of Questions, in D. M. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic, Second Edition*, pp. 1–60, Kluwer, Dordrecht/Boston/London.

Marius Hofert and Markus Kohm (2010), Scientific Presentations with LaTeX, *The PracTeX Journal*, 2010(2), `http://tug.org/pracjourn/2010-2/hofert.html`.

Paweł Łupkowski (2011), Human computation—how people solve difficult AI problems (having fun doing it), *Homo Ludens*, 3(1):81–94, ISSN 2080–4555.

Paweł Łupkowski (2015), Making your researcher's life easier. How to prepare transparent and dynamic research reports with LaTeX, in Tomasz Przechlewski, Karl Berry, Bogusław Jackowski, and Jerzy Ludwichowski, editors, *BachoTeX 2015: various faces of typography*, pp. 42–48, Polish TeX Users Group GUST, Bachotek.

Paweł Łupkowski (2016), *Logic of Questions in the Wild. Inferential Erotetic Logic in Information Seeking Dialogue Modelling*, College Publications, London.

Paweł Łupkowski and Jonathan Ginzburg (2013), A corpus-based taxonomy of question responses, in *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013)*, pp. 354–361, Association for Computational Linguistics, Potsdam, Germany, http://www.aclweb.org/anthology/W13-0209.

Paweł Łupkowski and Jonathan Ginzburg (2016), Query Responses, *Journal of Language Modelling*, 4(2):245–293.

Paweł Łupkowski and Olivia Ignaszak (2017), Inferential Erotetic Logic in Modelling of Cooperative Problem Solving Involving Questions in the QuestGen Game, *Organon F*, 24(2):214–244, http://www.klemens.sav.sk/fiusav/doc/organon/2017/2/214-244.pdf.

Paweł Łupkowski and Mariusz Urbański (2013), Preparing for scientific conferences with LaTeX: A short practical how-to, *TUGboat*, 34(2):184–189.

Paweł Łupkowski and Patrycja Wietrzycka (2015), Gamification for Question Processing Research—the QuestGen Game, *Homo Ludens*, 7(1):161–171.

Michal Peliš (2016), *Inferences with Ignorance: Logics of Questions (Inferential Erotetic Logic & Erotetic Epistemic Logic)*, Acta Universtitatis Carolinae – Philosophica et Historica, Karolinum, Praha.

R Core Team (2013), *R: A language and environment for statistical computing*, R Foundation for Statistical Computing, Vienna, Austria, http://www.R-project.org/, acess 20.03.2017.

Mariusz Urbański, Katarzyna Paluszkiewicz, and Joanna Urbańska (2016a), Erotetic Problem Solving: From Real Data to Formal Models. An Analysis of Solutions to Erotetic Reasoning Test Task, in Fabio Paglieri, Laura Bonetti, and Silvia Fellett, editors, *The Psychology of Argument: Cognitive Approaches to Argumentation and Persuasion*, pp. 33–46, College Publications, London.

Mariusz Urbański and Natalia Żyluk (2016), Sets of situations, topics, and question relevance, Technical report, AMU Institute of Psychology.

Mariusz URBAŃSKI, Natalia ŻYLUK, Katarzyna PALUSZKIEWICZ, and Joanna URBAŃSKA (2016b), A Formal Model of Erotetic Reasoning in Solving Some what Ill-Defined Problems, in D. MOHAMMED and M. LEWIŃSKI, editors, *Argumentation and Reasoned Action Proceedings of the 1st European Conference on Argumentation*, pp. 973–983, College Publications, London.

Jan VAN KUPPEVELT (1995), Discourse structure, topicality and questioning, *Journal of Linguistics*, 31:109–147.

Anthony J. VIERA and Joanne M. GARRETT (2005), Understanding Interobserver Agreement: The Kappa Statistic, *Family Medicine*, 37(5):360–363.

Luis VON AHN and Laura DABBISH (2008), Designing games with a purpose, *Communications of the ACM*, 51(8):58–67.

Andrzej WIŚNIEWSKI (1995), *The Posing of Questions: Logical Foundations of Erotetic Inferences*, Kluwer AP, Dordrecht, Boston, London.

Andrzej WIŚNIEWSKI (2013), *Questions, Inferences and Scenarios*, College Publications, London.

Andrzej WIŚNIEWSKI (2015), Semantics of Questions, in S. LAPPIN and Ch. FOX, editors, *The Handbook of Contemporary Semantic Theory, 2nd Edition*, pp. 273–313, Wiley-Blackwell, Oxford.