Filling gaps with Glue

Adam Przepiórkowski[1,2] and Agnieszka Patejuk[1]

[1]Institute of Computer Science, Polish Academy of Sciences

[2]Faculty of Philosophy, University of Warsaw

Proceedings of the LFG'23 Conference

Miriam Butt, Jamie Y. Findlay and Ida Toivonen (Editors)

2023

PubliKon

lfg-proceedings.org

Abstract

LFG currently lacks an analysis of gapping at the syntax-semantics interface. The aim of this paper is to fill this gap. We present two codescriptive approaches that do not assume empty constituents: one employing standard Glue but relying on a particular approach to event semantics, and another compatible with any approach to semantics but relying on a particular approach to Glue at the syntax-semantics interface.

1 Introduction

Consider the simple example of gapping (Ross 1970) in (1) and its intended neo-Davidsonian (Davidson 1967; Castañeda 1967; Parsons 1990) representation in (2), where the constants m, l, h, and b refer to *Marge*, *Lisa*, *Homer*, and *Bart*, respectively.[†] The representation in (2) reflects the fact that (1) refers to two seeing events, not just one.[1]

(1) Marge saw Lisa and Homer – Bart.

(2) $[\exists e.\, see(e) \wedge agent(e, m) \wedge theme(e, l)] \wedge$
$[\exists e.\, see(e) \wedge agent(e, h) \wedge theme(e, b)]$

The first bracketed conjunct in (2) faithfully represents the first clause of (1) (i.e., *Marge saw Lisa*). The challenge for any semantic theory of gapping is to – compositionally – obtain the second bracketed conjunct in (2) as the representation of the gapped clause in (1) (i.e., *Homer – Bart*), despite the missing verb.

Derivational approaches rely on there being a stage of derivation in which the eventually gapped clause does contain the lexical verb, i.e., a stage that can be represented as in (3).

(3) Marge saw Lisa and Homer saw Bart.

Subsequently, the second occurrence of *saw* is deleted or moved across-the-board out of the second conjunct. However, given that a copy or a trace of the verb remains in the gapped clause, deriving the right meaning representation of gapping is assumed to be straightforward in derivational syntax. See Park 2019: §4.2 for a useful overview of such approaches and discussion of problems that they face.

On the other hand, LFG eschews empty constituents unless there are good *syntactic* reasons to postulate them (Bresnan et al. 2016: §9.5), so the derivational approach is not applicable, and no other LFG-based account of the semantics of gapping has been proposed so far.

The aim of this paper is to propose two proof-of-concept codescriptive approaches to gapping relying on LFG syntax and Glue semantics (Dalrymple 1999; Asudeh 2022). Neither assumes empty constituents and both rely on the syntactic analysis of Patejuk & Przepiórkowski 2017. The first approach, presented in §2, employs standard Glue mechanisms, but it crucially relies on Champollion's

[†]We are grateful to LFG 2023 reviewers and audience for useful feedback.

[1]In this and other examples, the gap is explicitly marked with '–'.

(2015) compositional treatment of event semantics. The second approach, presented in §3, is compatible with any meaning representation language, but it relies on the XLE+Glue approach of Dalrymple et al. 2020 and on the possibility – currently *not* implemented within XLE (Crouch et al. 2011) – of treating certain attributes as "deeply distributive", i.e., on par with PRED. Conclusions are presented in §4.

2 An Event Semantics Approach

Glue is resource-sensitive, so the semantic representation see introduced by the predicate *saw* in (1) must be used exactly once in the glue proof. The question is how to obtain the representation in (2), repeated below, where see occurs twice.

(2) $[\exists e.\, \underline{see}(e) \wedge agent(e, m) \wedge theme(e, l)] \wedge$
$[\exists e.\, \underline{see}(e) \wedge agent(e, h) \wedge theme(e, b)]$

A somewhat similar problem occurs when a dependent introduces a resource that is shared between conjoined predicates, as in (4)–(5). The semantic representation of *Bart*, i.e., b, occurs twice in (5). The standard solution in such cases is that the representation of the shared dependent, b, is an argument of the representation of the coordination *walked and whistled* given in (6), resulting in the desired representation (5), as shown in (7).

(4) Bart walked and whistled.

(5) $[\exists e.\, walk(e) \wedge agent(e, \underline{b})] \wedge$
$[\exists e.\, whistle(e) \wedge agent(e, \underline{b})]$

(6) *walked and whistled* $\rightsquigarrow$
$\lambda x.\, [\exists e.\, walk(e) \wedge agent(e, x)] \wedge$
$[\exists e.\, whistle(e) \wedge agent(e, x)]$

(7) *Bart walked and whistled* $\rightsquigarrow$
$[\lambda x.\, [\exists e.\, walk(e) \wedge agent(e, x)] \wedge$
$[\exists e.\, whistle(e) \wedge agent(e, x)]](b) \quad \overset{\beta\text{-reduction}}{\rightsquigarrow} \quad (5)$

For a similar solution to work in gapping, the representation of the overt verb introduced in the first clause in (1) should be factored out and become an argument of the representation of the rest of the coordination, as schematically shown in (8):

(8) $[\lambda f.[\exists e.\, f(e) \wedge agent(e, m) \wedge theme(e, l)] \wedge$
$[\exists e.\, f(e) \wedge agent(e, h) \wedge theme(e, b)]](see) \quad \overset{\beta\text{-reduction}}{\rightsquigarrow} \quad (2)$

We flesh out this idea by building on Champollion's (2015) approach to event semantics (see §2.1) and on Patejuk & Przepiórkowski's (2017) LFG analysis of the syntax of gapping (see §2.2). An XLE+Glue (Dalrymple et al. 2020) implementation of this analysis is briefly described in §2.3, while §2.4 discusses the limitations of this approach.

2.1 Semantics

Champollion (2015) argues that, instead of the usual existential closure at the level of the clause, the existential quantifier which binds the event variable should be introduced by the verb itself, as in (9).

(9) *saw* $\rightsquigarrow \lambda f.\, \exists e.\, see(e) \wedge f(e)$

Dependents are analysed as semantic modifiers, as in (10)–(11), combining with verbs as in (12)–(13).

(10) *Marge*$_{agent}$ $\rightsquigarrow \lambda V.\lambda f.\, V(\lambda e.\, agent(e, m) \wedge f(e))$

(11) *Lisa*$_{theme}$ $\rightsquigarrow \lambda V.\lambda f.\, V(\lambda e.\, theme(e, l) \wedge f(e))$

(12) *saw Lisa* $\rightsquigarrow$ [(11)]((9)) $\overset{\beta\text{-reduction}}{\rightsquigarrow}$ $\lambda f.\, \exists e.\, see(e) \wedge theme(e, l) \wedge f(e)$

(13) *Marge saw Lisa* $\rightsquigarrow$ [(10)]((12)) $\overset{\beta\text{-reduction}}{\rightsquigarrow}$
$\lambda f.\, \exists e.\, see(e) \wedge theme(e, l) \wedge agent(e, m) \wedge f(e)$

The final relevant ingredient of Champollion 2015 is the event closure in (14) (replacing the usual existential closure), leading to the final representation of the sentence *Marge saw Lisa* in (15).

(14) [closure] $\rightsquigarrow \lambda e.\, true(e)$

(15) *Marge saw Lisa* (after closure) $\rightsquigarrow$ [(13)]((14)) $\overset{\beta\text{-reduction}}{\rightsquigarrow}$
$\exists e.\, see(e) \wedge theme(e, l) \wedge agent(e, m) \wedge true(e)$

Given that the predicate $true$ is always true, (15) is equivalent to the first bracketed conjunct in the representation of (1) given in (2) (both repeated below).

(1) Marge saw Lisa and Homer – Bart.

(2) $[\exists e.\, see(e) \wedge agent(e, m) \wedge theme(e, l)] \wedge$
$[\exists e.\, see(e) \wedge agent(e, h) \wedge theme(e, b)]$

For this approach to work in the analysis of gapping, semantic representations of verbs such as (9) must be split:

(16) *saw* $\rightsquigarrow \lambda V.\, \lambda f.\, V(\lambda e.\, see(e) \wedge f(e))$

(17) $\lambda f.\, \exists e.\, f(e)$

We assume that lexical verbs introduce two meaning constructors (MCs) corresponding to meaning representations in (16)–(17), while the gapped clause constructionally introduces a constructor corresponding to (17). In the case of the running example (1), when this MC combines with the agent *Homer* and the theme *Bart*, see (18)–(19), the representation of the gapped clause is (20):

(18) *Homer*$_{agent}$ $\rightsquigarrow \lambda V.\lambda f.\, V(\lambda e.\, agent(e, h) \wedge f(e))$

(19) *Bart*$_{theme}$ $\rightsquigarrow \lambda V.\lambda f.\, V(\lambda e.\, theme(e, b) \wedge f(e))$

(20) *Homer – Bart* (initial) $\rightsquigarrow$ [(18)]([(19)]((17))) $\overset{\beta\text{-reduction}}{\rightsquigarrow}$
$\lambda f.\, \exists e.\, theme(e, b) \wedge agent(e, h) \wedge f(e)$

Similarly, (21) is the initial representation of the first clause, utilizing only the MC corresponding to (17) (without (16)).

(21) *Marge saw Lisa* (initial) $\rightsquigarrow$ [(10)]([(11)]((17))) $\overset{\beta\text{-reduction}}{\rightsquigarrow}$
$\lambda f.\, \exists e.\, theme(e, l) \wedge agent(e, m) \wedge f(e)$

Assuming the standard approach to the semantics of coordination (Partee & Rooth 1983; Dalrymple et al. 2019: §16.7), (22) is the representation of *and* in the running example.

(22) *and* $\rightsquigarrow$ $\lambda V_1.\lambda V_2.\lambda f.\, V_1(f) \wedge V_2(f)$

Conjoining (21) and (20) results in (23):

(23) *Marge saw Lisa and Homer – Bart* (initial) $\rightsquigarrow$ [(22)]((21))((20)) $\overset{\beta\text{-reduction}}{\rightsquigarrow}$
$\lambda f.\, [\exists e.\, theme(e, l) \wedge agent(e, m) \wedge f(e)] \wedge$
$[\exists e.\, theme(e, b) \wedge agent(e, h) \wedge f(e)]$

Applying the representation of the verb in (16) to the representation of coordination in (23) results in (24):

(24) *Marge saw Lisa and Homer – Bart* (before closure) $\rightsquigarrow$ [(16)]((23)) $\overset{\beta\text{-reduction}}{\rightsquigarrow}$
$\lambda f.\, [\exists e.\, theme(e, l) \wedge agent(e, m) \wedge see(e) \wedge f(e)] \wedge$
$[\exists e.\, theme(e, b) \wedge agent(e, h) \wedge see(e) \wedge f(e)]$

This derivation is slightly more complex than the schematic derivation in (8), because in this derivation the representation of the verb, (16), is the functor rather than the argument. Applying (24) to the closure representation in (14) results in (25), which is equivalent to the desired representation (2) (repeated below again) of the running example.

(25) *Marge saw Lisa and Homer – Bart* (after closure) $\rightsquigarrow$ [(24)]((14)) $\overset{\beta\text{-reduction}}{\rightsquigarrow}$
$[\exists e.\, theme(e, l) \wedge agent(e, m) \wedge see(e) \wedge true(e)] \wedge$
$[\exists e.\, theme(e, b) \wedge agent(e, h) \wedge see(e) \wedge true(e)]$

(2) $[\exists e.\, see(e) \wedge agent(e, m) \wedge theme(e, l)] \wedge$
$[\exists e.\, see(e) \wedge agent(e, h) \wedge theme(e, b)]$

2.2 Syntax–Semantics Interface

On the analysis of gapping in Patejuk & Przepiórkowski 2017, c^2 in (26) is a partial f-structure of the gapped clause in (1).

(26)
$$c^2\begin{bmatrix} \textsc{subj} & \begin{bmatrix} \textsc{pred} & \text{`}\textsc{Homer}\text{'} \end{bmatrix} \\ \textsc{obj} & \begin{bmatrix} \textsc{pred} & \text{`}\textsc{Bart}\text{'} \end{bmatrix} \end{bmatrix}$$

A partial representation of the first (source) clause is more complex, as dependents that are not shared are put in the value of the LOCAL attribute, c^1, rather than in the main f-structure c; see (27).

(27)
$$c\begin{bmatrix} \textsc{pred} & \text{`}\textsc{see}\langle\textsc{subj, obj}\rangle\text{'} \\ \textsc{local} & c^1\begin{bmatrix} \textsc{subj} & \begin{bmatrix} \textsc{pred} & \text{`}\textsc{Marge}\text{'} \end{bmatrix} \\ \textsc{obj} & \begin{bmatrix} \textsc{pred} & \text{`}\textsc{Lisa}\text{'} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

The syntactic rule responsible for gapping is given in (28).

(28) IP → IP1 [Comma IP]* Conj IP
$\uparrow=\downarrow$ $\downarrow\in\uparrow$ $\downarrow\in\uparrow$
$(\downarrow \textsc{local}) \in \uparrow$

Given that IP1 corresponds to (27), and the last IP to (26), this rule results in (29) as the f-structure of the running example. Note that, according to this rule, c in (29) is analysed as a hybrid structure, so that PRED distributes to both conjuncts.[2]

(29)
$$c\left[\begin{array}{l}\left\{\begin{array}{l} c^1\left[\begin{array}{ll}\textsc{pred} & \text{`}\textsc{see}\langle\boxed{1},\boxed{2}\rangle\text{'}\\ \textsc{subj} & \boxed{1}\left[\textsc{pred}\quad \text{`}\textsc{Marge}\text{'}\right]\\ \textsc{obj} & \boxed{2}\left[\textsc{pred}\quad \text{`}\textsc{Lisa}\text{'}\right]\end{array}\right]\end{array}\right\},\ c^2\left[\begin{array}{ll}\textsc{pred} & \text{`}\textsc{see}\langle\boxed{3},\boxed{4}\rangle\text{'}\\ \textsc{subj} & \boxed{3}\left[\textsc{pred}\quad \text{`}\textsc{Homer}\text{'}\right]\\ \textsc{obj} & \boxed{4}\left[\textsc{pred}\quad \text{`}\textsc{Bart}\text{'}\right]\end{array}\right]\right\}\\ \textsc{local}\ \ c^1\\ \textsc{conj}\ \ \textsc{and}\end{array}\right]$$

As noted in §2.1, gapped clauses constructionally introduce an existentially bound event, $\lambda f.\,\exists e.\,f(e)$. This means that appropriate meaning constructors (MCs) must be added in the rule (28), as shown in (30).[3]

(30) IP → IP1 [Comma IP]* Conj IP
$\uparrow=\downarrow$ $\downarrow\in\uparrow$ $\downarrow\in\uparrow$
$(\downarrow \textsc{local}) \in \uparrow$ $\lambda f.\,\exists e.\,f(e) :$ $\lambda f.\,\exists e.\,f(e) :$
$(\downarrow_v \multimap \downarrow_t) \multimap \downarrow_t$ $(\downarrow_v \multimap \downarrow_t) \multimap \downarrow_t$

We assume that proper nouns introduce the usual MCs, e.g., $m : \uparrow_e$ for *Marge*. Thematic roles are introduced constructionally; for example, in a rule assigning the structure of a dependent to the SUBJ value of the verb, the dependent is also assumed to be an agent (perhaps as one of the options) and assigned the following MC:

(31) [agent] $\rightsquigarrow$
$\lambda x.\lambda V.\lambda f.\,V(\lambda e.\,agent(e,x) \wedge f(e)) :$
$\downarrow_e \multimap ((\uparrow_v \multimap \uparrow_t) \multimap \uparrow_t) \multimap (\uparrow_v \multimap \uparrow_t) \multimap \uparrow_t$

These two constructors, $m : \uparrow_e$ and (31), when combined in the analysis of the running example (1), result in the instantiated MC for the agent *Marge* in (32), where c^1 refers to the relevant f-structure in (29), and analogously for the theme *Lisa* in (33)–(34).

(32) $Marge_{agent} \rightsquigarrow$
$\lambda V.\lambda f.\,V(\lambda e.\,agent(e,m) \wedge f(e)) :\ ((c^1_v \multimap c^1_t) \multimap c^1_t) \multimap (c^1_v \multimap c^1_t) \multimap c^1_t$

(33) [theme] $\rightsquigarrow$
$\lambda x.\lambda V.\lambda f.\,V(\lambda e.\,theme(e,x) \wedge f(e)) :$
$\downarrow_e \multimap ((\uparrow_v \multimap \uparrow_t) \multimap \uparrow_t) \multimap (\uparrow_v \multimap \uparrow_t) \multimap \uparrow_t$

[2] We have more to say about this when we discuss the second analysis in §3.

[3] Throughout the paper, we assume the first-order approach to Glue (Kokkonidis 2008), with the usual basic semantic types e and t, as well as v – the type of events.

(34) $\mathit{Lisa}_{theme} \rightsquigarrow$
$\lambda V.\lambda f.\, V(\lambda e.\, theme(e,l) \wedge f(e)) :\ ((c_v^1 \multimap c_t^1) \multimap c_t^1) \multimap (c_v^1 \multimap c_t^1) \multimap c_t^1$

Similarly, the agent *Homer* and the theme *Bart* are associated with the MCs given in (35)–(36); in these cases $\uparrow$ instantiates to the f-structure of the gapped clause, i.e., to c^2.

(35) $\mathit{Homer}_{agent} \rightsquigarrow$
$\lambda V.\lambda f.\, V(\lambda e.\, agent(e,h) \wedge f(e)) :\ ((c_v^2 \multimap c_t^2) \multimap c_t^2) \multimap (c_v^2 \multimap c_t^2) \multimap c_t^2$

(36) $\mathit{Bart}_{theme} \rightsquigarrow$
$\lambda V.\lambda f.\, V(\lambda e.\, theme(e,b) \wedge f(e)) :\ ((c_v^2 \multimap c_t^2) \multimap c_t^2) \multimap (c_v^2 \multimap c_t^2) \multimap c_t^2$

The constructor introduced under the last IP in (30) (the gapped clause) in this case instantiates to (37) and it combines with MCs for the agent *Homer* and the theme *Bart* in (35)–(36), resulting in the MC (38) corresponding to the gapped clause.

(37) $\lambda f.\, \exists e.\, f(e) : (c_v^2 \multimap c_t^2) \multimap c_t^2$

(38) $\lambda f.\, \exists e.\, theme(e,b) \wedge agent(e,h) \wedge f(e) : (c_v^2 \multimap c_t^2) \multimap c_t^2$

Lexical verbs introduce the following three MCs – the first one, in (39), corresponds to the event closure (cf. (14) in §2.1), while the next two, in (40)–(41), correspond to the representations of the semantics of verbs (cf. (16)–(17) in §2.1):

(39) $\mathit{saw} \rightsquigarrow \lambda e.\, true(e) : \uparrow_v \multimap \uparrow_t$

(40) $\lambda V.\, \lambda f.\, V(\lambda e.\, see(e) \wedge f(e)) : ((\uparrow_v \multimap \uparrow_t) \multimap \uparrow_t) \multimap (\uparrow_v \multimap \uparrow_t) \multimap \uparrow_t$

(41) $\lambda f.\, \exists e.\, f(e) : (\%\mathrm{v}_v \multimap \%\mathrm{v}_t) \multimap \%\mathrm{v}_t, \quad \text{where} \quad \%\mathrm{v} = (\uparrow (\textsc{local}))$

Note that (41) relies on the variable $\%\mathrm{v}$ which can resolve to $\uparrow$ or $(\uparrow \textsc{local})$. The latter is used in the analysis of gapping: in the running example (1) with the f-structure in (29), (41) instantiates to (42) and combines with the agent *Marge* and the theme *Lisa* in a way analogous to the gapped clause, resulting in the MC (43) corresponding to the first conjunct.

(42) $\mathit{saw} \rightsquigarrow \lambda f.\, \exists e.\, f(e) : (c_v^1 \multimap c_t^1) \multimap c_t^1$

(43) $\lambda f.\, \exists e.\, theme(e,l) \wedge agent(e,m) \wedge f(e) : (c_v^1 \multimap c_t^1) \multimap c_t^1$

The resulting MCs (43) and (38) for the two conjuncts are arguments to the instantiation of the standard meaning of *and* given in (44) (cf. (22) in §2.1); note that the arguments correspond to conjuncts c^1 and c^2, and the result corresponds to the coordination c.

(44) $\lambda V_1.\lambda V_2.\lambda f.\, V_1(f) \wedge V_2(f) :$
$((c_v^1 \multimap c_t^1) \multimap c_t^1) \multimap ((c_v^2 \multimap c_t^2) \multimap c_t^2) \multimap (c_v \multimap c_t) \multimap c_t$

When (44) is applied to the MCs of conjuncts in (43) and (38), the result is (45) below (cf. (23) in §2.1).

(45) $\lambda f.\, [\exists e.\, theme(e,l) \wedge agent(e,m) \wedge f(e)] \wedge$
$[\exists e.\, theme(e,b) \wedge agent(e,h) \wedge f(e)] : (c_v \multimap c_t) \multimap c_t$

In the other – idiosyncratic – MC introduced by the verb, (40), $\uparrow$ is instantiated to c, as shown in (46).

(46) *saw* $\rightsquigarrow \lambda V.\lambda f.V(\lambda e.see(e) \wedge f(e)) : ((c_v \multimap c_t) \multimap c_t) \multimap (c_v \multimap c_t) \multimap c_t$

The MC in (45) is input to the MC in (46), resulting in (47) (cf. (24) in §2.1):

(47) $\lambda f.[\exists e.theme(e,l) \wedge agent(e,m) \wedge see(e) \wedge f(e)] \wedge$
$[\exists e.theme(e,b) \wedge agent(e,h) \wedge see(e) \wedge f(e)] : (c_v \multimap c_t) \multimap c_t$

Finally, applying (47) to the closure in (39) instantiated as in (48) results in the final representation in (49) (cf. (25) and (2) in §2.1).

(48) *saw* $\rightsquigarrow \lambda e.true(e) : c_v \multimap c_t$

(49) $[\exists e.theme(e,l) \wedge agent(e,m) \wedge see(e) \wedge true(e)] \wedge$
$[\exists e.theme(e,b) \wedge agent(e,h) \wedge see(e) \wedge true(e)] : c_t$

2.3 Implementation

The analysis presented above has been computationally verified as an XLE+Glue (Dalrymple et al. 2020) implementation.

The c-structure tree for the running example (1) (repeated below again) is given in Figure 1, the simplified f-structure in Figure 2, and the full f-structure in Figure 3.

(1) Marge saw Lisa and Homer – Bart.

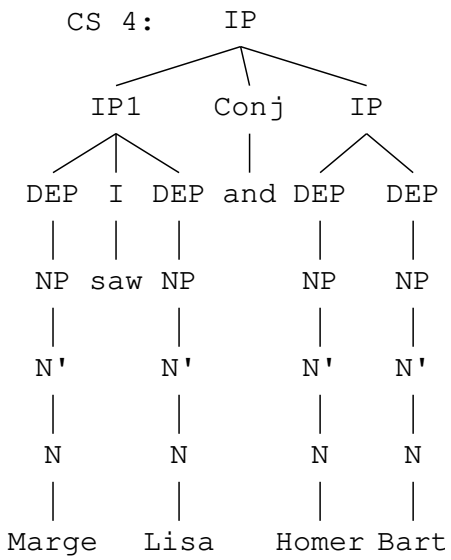

Figure 1: C-structure for (1) generated by the XLE+Glue implementation

Figure 3 illustrates the actual encoding of MCs in XLE+Glue. All such MCs – values of the attributes GLUE – are collected and transferred to a linear theorem prover (Glue Semantics Workbench, GSWB; Meßmer & Zymla 2018). Multiple proofs

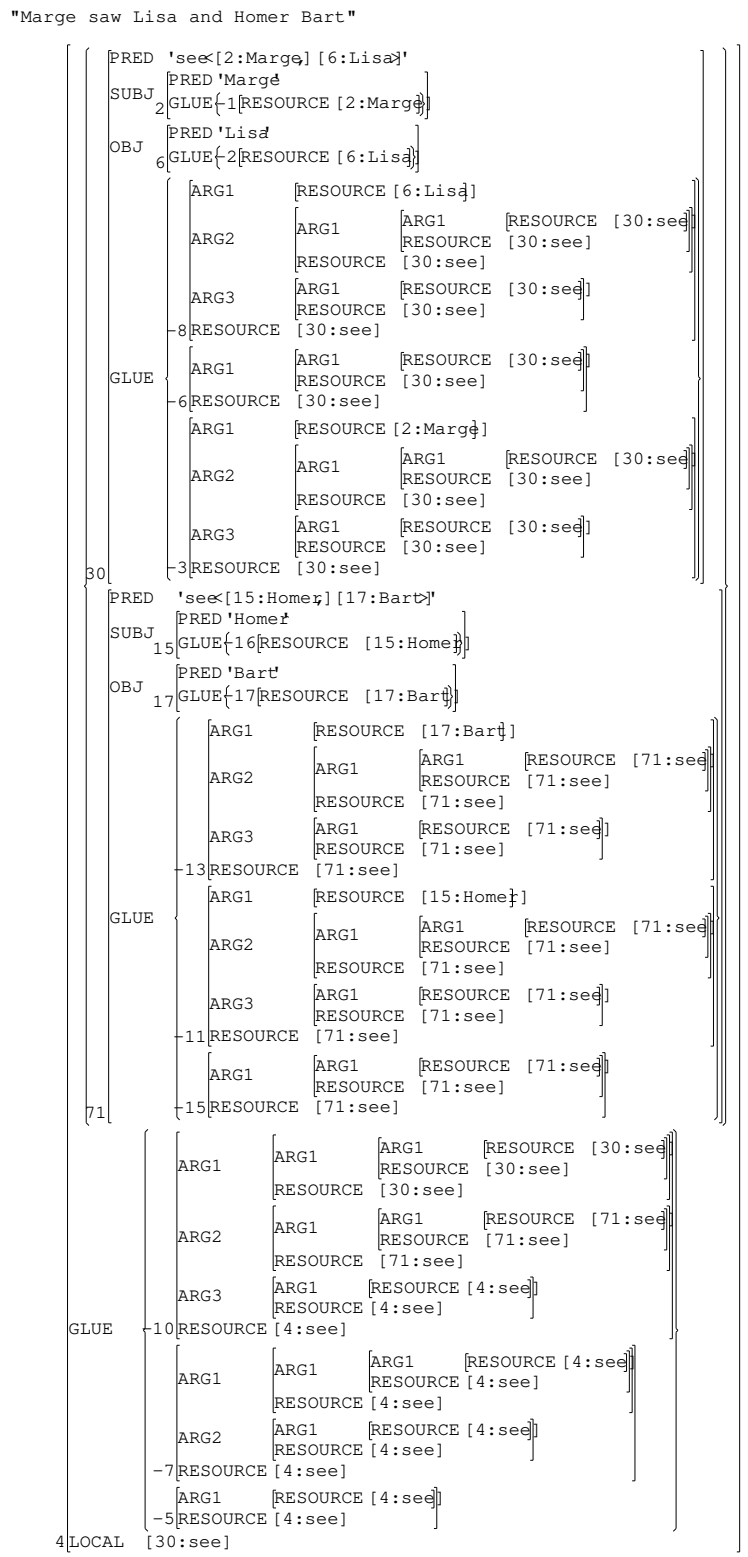

Figure 2: Simplified f-structure for (1) generated by the XLE+Glue implementation

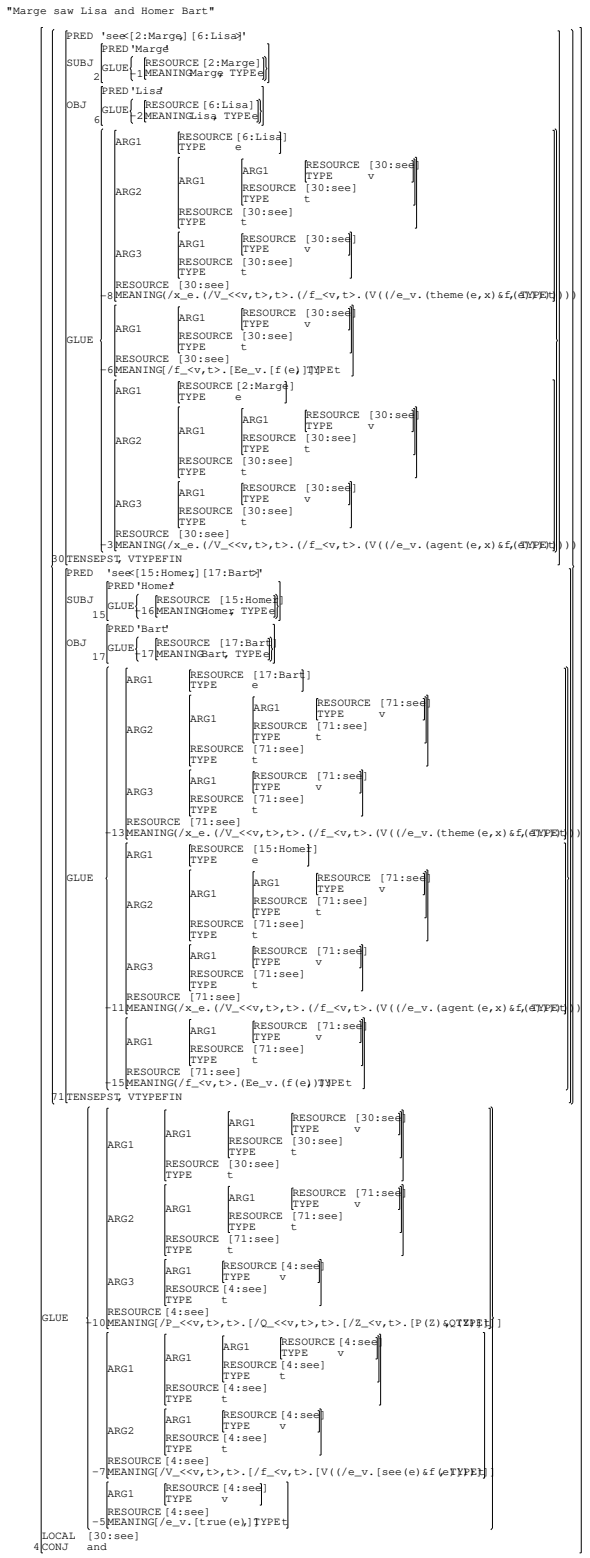

Figure 3: Full f-structure for (1) generated by the XLE+Glue implementation

result in four different meaning representations, all equivalent to (2) (repeated below). As shown in (50)–(53), they only differ in the order of conjuncts, reflecting the order in which meaning representations combine.[4]

(2) $[\exists e.\, see(e) \wedge agent(e, m) \wedge theme(e, l)] \wedge$
$[\exists e.\, see(e) \wedge agent(e, h) \wedge theme(e, b)]$

(50) $\exists$`e2[agent(e2,Marge)` $\wedge$ `theme(e2,Lisa)` $\wedge$ `see(e2)` $\wedge$ `true(e2)]` $\wedge$
$\exists$`e1[agent(e1,Homer)` $\wedge$ `theme(e1,Bart)` $\wedge$ `see(e1)` $\wedge$ `true(e1)]`

(51) $\exists$`e2[agent(e2,Marge)` $\wedge$ `theme(e2,Lisa)` $\wedge$ `see(e2)` $\wedge$ `true(e2)]` $\wedge$
$\exists$`e1[theme(e1,Bart)` $\wedge$ `agent(e1,Homer)` $\wedge$ `see(e1)` $\wedge$ `true(e1)]`

(52) $\exists$`e2[theme(e2,Lisa)` $\wedge$ `agent(e2,Marge)` $\wedge$ `see(e2)` $\wedge$ `true(e2)]` $\wedge$
$\exists$`e1[agent(e1,Homer)` $\wedge$ `theme(e1,Bart)` $\wedge$ `see(e1)` $\wedge$ `true(e1)]`

(53) $\exists$`e2[theme(e2,Lisa)` $\wedge$ `agent(e2,Marge)` $\wedge$ `see(e2)` $\wedge$ `true(e2)]` $\wedge$
$\exists$`e1[theme(e1,Bart)` $\wedge$ `agent(e1,Homer)` $\wedge$ `see(e1)` $\wedge$ `true(e1)]`

The implementation also verifies that the above analysis correctly deals with gapping involving dependents shared between the conjuncts, as in (54), which gives rise to two different semantic representations in (55)–(56) (differences between them are underlined).

(54) Tracy introduced Lisa to Marge and Bart to Homer.

(55) $[\exists e.\, introduce(e) \wedge agent(e, t) \wedge theme(e, l) \wedge beneficiary(e, m)] \wedge$
$[\exists e.\, introduce(e) \wedge \underline{agent(e, t)} \wedge \underline{theme(e, b)} \wedge beneficiary(e, h)]$
'Tracy introduced Lisa to Marge and Tracy introduced Bart to Homer.'

(56) $[\exists e.\, introduce(e) \wedge agent(e, t) \wedge theme(e, l) \wedge beneficiary(e, m)] \wedge$
$[\exists e.\, introduce(e) \wedge \underline{agent(e, b)} \wedge \underline{theme(e, l)} \wedge beneficiary(e, h)]$
'Tracy introduced Lisa to Marge and Bart introduced Lisa to Homer.'

2.4 Limitations

The crucial assumption of this analysis is that verbs do not lexically refer to their arguments in their semantic representations. This assumption is met by Champollion's (2015) representation in (9), translated into the MCs in (40)–(41) (all repeated below).

(9) *saw* $\rightsquigarrow \lambda f.\, \exists e.\, see(e) \wedge f(e)$

(40) *saw* $\rightsquigarrow \lambda V.\, \lambda f.\, V(\lambda e.\, see(e) \wedge f(e)) : ((\uparrow_v \multimap \uparrow_t) \multimap \uparrow_t) \multimap (\uparrow_v \multimap \uparrow_t) \multimap \uparrow_t$

(41) $\lambda f.\, \exists e.\, f(e) : (\%\mathrm{v}_v \multimap \%\mathrm{v}_t) \multimap \%\mathrm{v}_t$, where $\%\mathrm{v} = (\uparrow (\textsc{local}))$

An MC analogous to (41) is also introduced constructionally in gapped clauses, namely, in rule (30) (repeated below).

[4] Findlay & Haug 2022 propose a way to deal with some kinds of spurious ambiguities resulting from Glue proofs, but it does not seem to be directly applicable to the case at hand.

(30) IP $\rightarrow$ IP1 [Comma IP]* Conj IP

	$\uparrow{=}\downarrow$		$\downarrow{\in}\uparrow$	$\downarrow{\in}\uparrow$
	$(\downarrow \textsc{local}) \in \uparrow$		$\lambda f.\, \exists e.\, f(e) :$	$\lambda f.\, \exists e.\, f(e) :$
			$(\downarrow_v \multimap \downarrow_t) \multimap \downarrow_t$	$(\downarrow_v \multimap \downarrow_t) \multimap \downarrow_t$

Importantly, the MCs in (30) do not assume any particular number of dependents – "remnants" – in the gapped clause(s) or what grammatical functions these remnants bear. In the case of the running example (1) there are two remnants, a subject and an object, but gapped clauses may consist of a larger number of remnants and they can bear different grammatical functions. For example, there are four remnants in (57) (Sag 1976: 278, ex. (3.4.51)) – a subject, an oblique dependent expressing accompaniment, and two adjuncts (locative and temporal).

(57) Betsy dances with a parasol in the living room on Fridays and Peter with a meat cleaver in the bar on Saturday nights.

Also the (instantiated) MC for the conjunction *and*, given in (44) and repeated below, does not mention the number or kind of remnants.

(44) $\lambda V_1.\lambda V_2.\lambda f.\, V_1(f) \wedge V_2(f) :$
$((c_v^1 \multimap c_t^1) \multimap c_t^1) \multimap ((c_v^2 \multimap c_t^2) \multimap c_t^2) \multimap (c_v \multimap c_t) \multimap c_t$

What would not work as smoothly is a representation such as (58) or (59), where arguments are referred to directly.

(58) *saw* $\rightsquigarrow$ $\lambda x.\lambda y.\, see(x, y)$

(59) *saw* $\rightsquigarrow$ $\lambda x.\lambda y.\, \lambda e.\, see(e) \wedge agent(e, x) \wedge theme(e, y)$

We will illustrate the problem on the basis of the simpler representation (58).

At the level of pure meaning representations, without the linear Glue part, a solution seems to be possible that is fully analogous to the analysis proposed above. First, we may split (58) into two representations, one of which is also introduced constructionally at the level of the gapped clause:

(60) *saw* $\rightsquigarrow$ $\lambda x.\lambda y.\, see(x, y)$

(61) $\lambda x.\lambda y.\lambda f.\, f(x, y)$

(62) gapped clause $\rightsquigarrow$ $\lambda x.\lambda y.\lambda f.\, f(x, y)$

The representations in (61) and (62) would then combine with the representations of arguments within the two conjuncts, giving rise to (63)–(64):

(63) $\lambda f.\, f(m, l)$

(64) $\lambda f.\, f(h, b)$

These can be coordinated, assuming a variant of the usual representation of the conjunction in (65), resulting in (66).

(65) $\lambda V_1.\lambda V_2.\lambda f.\, V_1(f) \wedge V_2(f)$

(66) $\lambda f.\, f(m, l) \wedge f(h, b)$

The resulting representation (66) can then be applied to the idiosyncratic representation introduced by the verb in (60), resulting in the desired (67):

(67) $see(m, l) \wedge see(h, b)$

However, the problem becomes clear once we introduce complete MCs corresponding to (60)–(62):

(68) *saw* $\rightsquigarrow$ $\lambda x.\lambda y.\, see(x, y) : (\%\text{V SUBJ})_e \multimap (\%\text{V OBJ})_e \multimap \%\text{V}_t$

(69) $\lambda x.\lambda y.\lambda f.\, f(x, y) :$
$(\%\text{V SUBJ})_e \multimap (\%\text{V OBJ})_e \multimap ((\%\text{V SUBJ})_e \multimap (\%\text{V OBJ})_e \multimap \%\text{V}_t) \multimap \%\text{V}_t$

(70) gapped clause $\rightsquigarrow$
$\lambda x.\lambda y.\lambda f.\, f(x, y) :$
$(\uparrow \text{SUBJ})_e \multimap (\uparrow \text{OBJ})_e \multimap ((\uparrow \text{SUBJ})_e \multimap (\uparrow \text{OBJ})_e \multimap \uparrow_t) \multimap \uparrow_t$

In (68)–(69), %V is equal to $\uparrow$ or $(\uparrow \text{LOCAL})$, as in (41) above.[5] Given the syntactic structure of gapping assumed above, these MCs are instantiated as follows:

(71) *saw* $\rightsquigarrow$ $\lambda x.\lambda y.\, see(x, y) : (c^1\ \text{SUBJ})_e \multimap (c^1\ \text{OBJ})_e \multimap c^1_t$

(72) $\lambda x.\lambda y.\lambda f.\, f(x, y) :$
$(c^1\ \text{SUBJ})_e \multimap (c^1\ \text{OBJ})_e \multimap ((c^1\ \text{SUBJ})_e \multimap (c^1\ \text{OBJ})_e \multimap c^1_t) \multimap c^1_t$

(73) gapped clause $\rightsquigarrow$
$\lambda x.\lambda y.\lambda f.\, f(x, y) :$
$(c^2\ \text{SUBJ})_e \multimap (c^2\ \text{OBJ})_e \multimap ((c^2\ \text{SUBJ})_e \multimap (c^2\ \text{OBJ})_e \multimap c^2_t) \multimap c^2_t$

The problem is that such meaning constructors, including the constructionally introduced MC for the gapped clause in (70), have to assume not only a specific number of dependents, but also their grammatical functions. Hence, in order for the solution to be general, a large disjunction of MCs such as (70) would have to be introduced constructionally: different combinations of grammatical functions for each number of dependents, with an arbitrary cutoff point on the maximal number of dependents that can occur in gapping.

Also a large number of MCs for conjunctions would be needed. The deceptively simple representation (65) corresponds to the instantiated MC in (74) needed for the running example, in which there are two remnants, a subject and an object:

(74) $\lambda V_1.\lambda V_2.\lambda f.\, V_1(f) \wedge V_2(f) :$
$(((c^1\ \text{SUBJ})_e \multimap (c^1\ \text{OBJ})_e \multimap c^1_t) \multimap c^1_t) \multimap$
$(((c^2\ \text{SUBJ})_e \multimap (c^2\ \text{OBJ})_e \multimap c^2_t) \multimap c^2_t) \multimap$
$((c^1\ \text{SUBJ})_e \multimap (c^1\ \text{OBJ})_e \multimap c^1_t) \multimap c_t$

Again, such MCs for *and* would have to be added for any number of dependents (up to an arbitrary cutoff point) and for any combination of grammatical functions.

Apart from the diminished readability of grammars with such a large number

[5] In (39)–(41) above, which present MCs for *saw* in the analysis based on Champollion's (2015) approach, only one MC, in (41), refers to %V. By contrast, both MCs (68)–(69) introduced by *saw* are based on %V rather than $\uparrow$. This is because, in gapping constructions, $\uparrow$ instantiates to the hybrid structure c, which does not directly contain grammatical function attributes such as SUBJ or OBJ, so there is no specific resource corresponding to, say, $(\uparrow \text{SUBJ})$. On the other hand, %V instantiates in such constructions to the non-gapped clause c^1, which does contain such grammatical function attributes, so there is a resource corresponding to, say, $(\%\text{V SUBJ})$.

of MCs, this solution may be too inefficient to be implementable in XLE+Glue.[6] Hence, the solution of the preceding subsections is practically, if not theoretically, limited to Champollion's (2015) approach to event semantics or a variant thereof.

We do not consider this a serious limitation, as there are well-known independent arguments for adopting event semantics in general (Parsons 1990) and for Champollion's implementation of composition in event semantics in particular (Champollion 2015). Nevertheless, this is a limitation that the approach described below attempts to avoid.

3 A "Deep Distributivity" Approach

An attempt at a more general alternative solution is inspired by an idea implemented in XLE+Glue, namely, to encode meaning constructors as AVMs. We follow XLE+Glue in assuming that such AVMs are members of set-valued GLUE attributes within f-structures, though they could be placed in a separate projection. The specific encoding of MCs as AVMs will not concern us here (but see Figure 3 for an example); the details may be found in Dalrymple et al. 2020. To simplify, we will represent such AVM MCs as string MCs in scare quotes, e.g.: '$m : \uparrow_e$'. For example, partial lexical entries for *Marge* and *saw* may be represented as in (75)–(76), with the semantic contribution of *saw* encoded as the simple, eventless $\lambda x.\lambda y.\, see(x, y)$.

(75) *Marge* N $(\uparrow \textsc{pred}) = \text{'}\textsc{Marge}\text{'}$
'$m : \uparrow_e$' $\in$ $(\uparrow \textsc{glue})$

(76) *saw* V $(\uparrow \textsc{pred}) = \text{'}\textsc{see}\langle(\uparrow \textsc{subj}), (\uparrow \textsc{obj})\rangle\text{'}$
'$\lambda x.\lambda y.\, see(x, y) : (\uparrow \textsc{subj})_e \multimap (\uparrow \textsc{obj})_e \multimap \uparrow_t$' $\in$ $(\uparrow \textsc{glue})$

Assuming the lexical entry for *Lisa* in (77), fully analogous to that for *Marge* in (75), the simple sentence (78) receives the f-structure (79).

(77) *Lisa* N $(\uparrow \textsc{pred}) = \text{'}\textsc{Lisa}\text{'}$
'$l : \uparrow_e$' $\in$ $(\uparrow \textsc{glue})$

(78) Marge saw Lisa.

(79)
$$\begin{bmatrix}
\textsc{pred} & \text{'}\textsc{see}\text{<}(f\ \textsc{subj}), (f\ \textsc{obj})\text{>}\text{'} \\
\textsc{subj} & s\begin{bmatrix} \textsc{pred} & \text{'}\textsc{Marge}\text{'} \\ \textsc{glue} & \{\text{'}m : s_e\text{'}\} \end{bmatrix} \\
\textsc{obj} & o\begin{bmatrix} \textsc{pred} & \text{'}\textsc{Lisa}\text{'} \\ \textsc{glue} & \{\text{'}l : o_e\text{'}\} \end{bmatrix} \\
\textsc{glue} & \{\text{'}\lambda x.\lambda y.\, see(x, y) : (f\ \textsc{subj})_e \multimap (f\ \textsc{obj})_e \multimap f_t\text{'}\}
\end{bmatrix}$$

[6] One way to alleviate this problem to some extent would be to define an abbreviation, GF, for a disjunction of relevant grammatical functions, $\textsc{gf} \equiv \{\textsc{subj}|\textsc{obj}|\dots\}$, and use it in such MCs – together with local names – instead of specific grammatical functions. For example, (70) could be generalized to (i), where $\%\textsc{x} = (\uparrow \textsc{gf})$ and $\%\textsc{y} = (\uparrow \textsc{gf})$.

(i) gapped clause $\rightsquigarrow$ $\lambda x.\lambda y.\lambda f.\, f(x, y) : \%\textsc{x}_e \multimap \%\textsc{y}_e \multimap (\%\textsc{x}_e \multimap \%\textsc{y}_e \multimap \uparrow_t) \multimap \uparrow_t$

However, it would still be necessary to have a disjunction of such generalized MCs for different numbers of remnants.

As mentioned above, in XLE+Glue all values of the attributes GLUE are collected and transferred to a linear theorem prover. In the case of example (78) and its f-structure in (79), this results in a proof of the desired meaning representation in (80).

(80) $see(m, l) : f_t$

The analysis of gapping proposed below relies on the possibility – currently *not* implemented in XLE – to define GLUE as "deeply distributive", in the sense in which PRED is hardcoded in XLE to be "deeply distributive". Let us illustrate "deep distributivity" of PRED with an example.

Assume the structure c partially specified as in (81). Combined with the equation in (82), c has the structure in (83).

(81) $c = \left\{ \begin{bmatrix} \textsc{subj} & \begin{bmatrix} \textsc{pred} & \text{`Marge'} \end{bmatrix} \\ \textsc{obj} & \begin{bmatrix} \textsc{pred} & \text{`Lisa'} \end{bmatrix} \end{bmatrix}, \begin{bmatrix} \textsc{subj} & \begin{bmatrix} \textsc{pred} & \text{`Homer'} \end{bmatrix} \\ \textsc{obj} & \begin{bmatrix} \textsc{pred} & \text{`Bart'} \end{bmatrix} \end{bmatrix} \right\}$

(82) $(c\ \textsc{pred}) = \text{`}\textsc{see}\langle(c\ \textsc{subj}), (c\ \textsc{obj})\rangle\text{'}$

(83) $c = \left\{ \begin{bmatrix} \textsc{pred} & \text{`}\textsc{see}\text{<}\boxed{1},\boxed{2}\text{>'} \\ \textsc{subj} & \boxed{1}\begin{bmatrix} \textsc{pred} & \text{`Marge'} \end{bmatrix} \\ \textsc{obj} & \boxed{2}\begin{bmatrix} \textsc{pred} & \text{`Lisa'} \end{bmatrix} \end{bmatrix}, \begin{bmatrix} \textsc{pred} & \text{`}\textsc{see}\text{<}\boxed{3},\boxed{4}\text{>'} \\ \textsc{subj} & \boxed{3}\begin{bmatrix} \textsc{pred} & \text{`Homer'} \end{bmatrix} \\ \textsc{obj} & \boxed{4}\begin{bmatrix} \textsc{pred} & \text{`Bart'} \end{bmatrix} \end{bmatrix} \right\}$

What is important here is that the specifications $(c\ \textsc{subj})$ and $(c\ \textsc{obj})$ in (82) are resolved independently for each conjunct, resulting in two different values of PRED in (83); hence "deep distributivity". The syntactic analysis in Patejuk & Przepiórkowski 2017 relies on this behaviour of PRED.

Returning to structures with the attribute GLUE specified as in the lexical entries (75)–(77), the syntactic part of the analysis of gapping produces the f-structures (84)–(85), fully analogous to (26)–(27) above. If the attribute GLUE could be made to behave like PRED, the verb's GLUE in (85) would distribute to conjuncts and specifications such as $(c\ \textsc{subj})$, etc., would be resolved independently to $(c^1\ \textsc{subj})$ (i.e., to $\boxed{1}$) and to $(c^2\ \textsc{subj})$ (i.e., to $\boxed{3}$), etc., just as they do in the values of PRED. This desired effect is shown in (86).[7]

[7] Making GLUE distributive requires a non-standard representation of conjunctions (see (86)).

Under the standard analysis, shown in (i), the CONJ attribute has an atomic value which hosts the conjunction form/type (AND, OR, etc.), while the corresponding GLUE attribute is placed at the same level as CONJ and the set containing conjuncts. Under this analysis, when GLUE is made distributive, the meaning constructor of the conjunction would be distributed (and so multiplied), which is undesired.

(i) $\begin{bmatrix} \left\{ \left[\dots\right], \left[\dots\right] \right\} \\ \textsc{conj} \quad \textsc{and} \\ \textsc{glue} \quad \left\{\dots\right\} \end{bmatrix}$ (ii) $\begin{bmatrix} \left\{ \left[\dots\right], \left[\dots\right] \right\} \\ \textsc{conj} \quad \begin{bmatrix} \textsc{form} & \textsc{and} \\ \textsc{glue} & \left\{\dots\right\} \end{bmatrix} \end{bmatrix}$

This is why distributive GLUE requires an alternative representation of the conjunction, shown in (ii), where the value of the CONJ attribute is an f-structure containing a FORM attribute whose value corresponds to the conjunction form/type and a GLUE attribute containing the meaning constructor of the conjunction. Since CONJ is non-distributive, the meaning constructor of the conjunction will not be distributed (and so multiplied) under this modified analysis, as desired.

(84)
$$c^2\left[\begin{array}{ll}\textsc{subj} & \boxed{3}\left[\begin{array}{ll}\textsc{pred} & \text{`}\textsc{Homer}\text{'}\\ \textsc{glue} & \left\{\text{`}h:\boxed{3}_e\text{'}\right\}\end{array}\right]\\ \textsc{obj} & \boxed{4}\left[\begin{array}{ll}\textsc{pred} & \text{`}\textsc{Bart}\text{'}\\ \textsc{glue} & \left\{\text{`}b:\boxed{4}_e\text{'}\right\}\end{array}\right]\end{array}\right]$$

(85)
$$c\left[\begin{array}{ll}\textsc{pred} & \text{`}\textsc{see}\text{<}(c\ \textsc{subj}),(c\ \textsc{obj})\text{>'}\\ \textsc{glue} & \left\{\text{`}\lambda x.\lambda y.\,see(x,y):(c\ \textsc{subj})_e\multimap(c\ \textsc{obj})_e\multimap c_t\text{'}\right\}\\ \textsc{local} & c^1\left[\begin{array}{ll}\textsc{subj} & \boxed{1}\left[\begin{array}{ll}\textsc{pred} & \text{`}\textsc{Marge}\text{'}\\ \textsc{glue} & \left\{\text{`}m:\boxed{1}_e\text{'}\right\}\end{array}\right]\\ \textsc{obj} & \boxed{2}\left[\begin{array}{ll}\textsc{pred} & \text{`}\textsc{Lisa}\text{'}\\ \textsc{glue} & \left\{\text{`}l:\boxed{2}_e\text{'}\right\}\end{array}\right]\end{array}\right]\end{array}\right]$$

(86)
$$c\left[\begin{array}{l}\left\{\begin{array}{l}c^1\left[\begin{array}{ll}\textsc{pred} & \text{`}\textsc{see}\text{<}\boxed{1},\boxed{2}\text{>'}\\ \textsc{glue} & \left\{\text{`}\lambda x.\lambda y.\,see(x,y):\boxed{1}_e\multimap\boxed{2}_e\multimap c_t^1\text{'}\right\}\\ \textsc{subj} & \boxed{1}\left[\begin{array}{ll}\textsc{pred} & \text{`}\textsc{Marge}\text{'}\\ \textsc{glue} & \left\{\text{`}m:\boxed{1}_e\text{'}\right\}\end{array}\right]\\ \textsc{obj} & \boxed{2}\left[\begin{array}{ll}\textsc{pred} & \text{`}\textsc{Lisa}\text{'}\\ \textsc{glue} & \left\{\text{`}l:\boxed{2}_e\text{'}\right\}\end{array}\right]\end{array}\right],\\ c^2\left[\begin{array}{ll}\textsc{pred} & \text{`}\textsc{see}\text{<}\boxed{3},\boxed{4}\text{>'}\\ \textsc{glue} & \left\{\text{`}\lambda x.\lambda y.\,see(x,y):\boxed{3}_e\multimap\boxed{4}_e\multimap c_t^2\text{'}\right\}\\ \textsc{subj} & \boxed{3}\left[\begin{array}{ll}\textsc{pred} & \text{`}\textsc{Homer}\text{'}\\ \textsc{glue} & \left\{\text{`}h:\boxed{3}_e\text{'}\right\}\end{array}\right]\\ \textsc{obj} & \boxed{4}\left[\begin{array}{ll}\textsc{pred} & \text{`}\textsc{Bart}\text{'}\\ \textsc{glue} & \left\{\text{`}b:\boxed{4}_e\text{'}\right\}\end{array}\right]\end{array}\right]\end{array}\right\}\\ \textsc{local}\ \ c^1\\ \textsc{conj}\ \ \left[\begin{array}{ll}\textsc{form} & \textsc{and}\\ \textsc{glue} & \left\{\text{`}\lambda p.\lambda q.\,p\wedge q:c_t^1\multimap c_t^2\multimap c_t\text{'}\right\}\end{array}\right]\end{array}\right]$$

Values of the GLUE attributes in this f-structure give rise to a proof of the desired meaning representation in (87).

(87) $see(m,l)\wedge see(h,b):c_t$

More generally, the analysis proposed here leads to the correct account of the semantics of gapping, regardless of the meaning representation assumed, i.e., without the need to assume event semantics.

An interesting feature of this analysis is the multiplication of Glue resources outside of the pure Glue system: while the MC which is the value of the verb's GLUE attribute is introduced just once, in the lexical entry (76), it gets multiplied via the syntactic mechanism of f-structure distributivity, as illustrated in (86), due to the distributivity of the attribute GLUE. The fact that syntax impinges on the resource sensitivity of Glue could be considered as a potential conceptual problem, but we believe that such behaviour is justified and desired: if PRED is allowed to distribute to conjuncts, a similar distribution of semantic aspects defined in GLUE should not be controversial.

Unfortunately, as noted above, there does not seem to be a way of making GLUE behave like PRED in the current implementation of XLE, i.e., behave in such a way that the ↑ metavariable used in assignments of GLUE values is instantiated independently in each conjunct. As verified implementationally, even though GLUE can be declared as a distributive attribute, ↑ used in GLUE points to the whole coordinate structure c (instead of resolving to c^1 in the first conjunct and c^2 in the second), while (↑ SUBJ) gets resolved to the subsumption of the SUBJ values of the two conjuncts (and similarly for (↑ OBJ)). What is needed is a generalization of the XLE treatment of PRED, i.e., a mechanism to declare an arbitrary attribute as similarly "deeply distributive".

4 Conclusion

To the best of our knowledge, there are no previous LFG analyses of gapping at the syntax-semantics interface, and this paper seeks to change this situation. We present two codescriptive analyses that build on the syntactic account of gapping proposed in Patejuk & Przepiórkowski 2017. The first analysis relies on Champollion's (2015) approach to event semantics but otherwise does not make any non-standard assumptions. It is based on standard theoretical Glue mechanisms and it has been verified via a relatively straightforward XLE+Glue implementation. The other analysis does not rely on event semantics – it is compatible with any meaning representation language. However, it relies on the AVM encoding of meaning constructors within f-structures, as proposed in Dalrymple et al. 2020, and – crucially – on the possibility of making the relevant f-structure attribute (namely, GLUE) "deeply distributive", on par with PRED. As this last possibility is not currently implemented in XLE, this analysis has not been implementationally verified.

Both analyses have only been applied to simple gapping constructions, involving coordination and a single finite verb missing – perhaps together with some dependents – from the gapped clause. An extension of this analysis to more complex constructions, involving missing verb chains, subconstituents, and occurrences of gapping outside of coordination, discussed in Park 2019 and elsewhere, is left for future work.

References

Asudeh, Ash. 2022. Glue semantics. *Annual Review of Linguistics* 8. 321–341. DOI: https://doi.org/10.1146/annurev-linguistics-032521-053835.

Bresnan, Joan, Ash Asudeh, Ida Toivonen & Stephen Wechsler. 2016. *Lexical-functional syntax* Blackwell Textbooks in Linguistics. Wiley-Blackwell 2nd edn.

Castañeda, Hector Neri. 1967. Comment on D. Davidson's 'The logical form of action sentences'. In Rescher (1967) 104–112.

Champollion, Lucas. 2015. The interaction of compositional semantics and event semantics. *Linguistics and Philosophy* 38(1). 31–66. DOI: https://doi.org/10.1007/s10988-014-9162-8.

Crouch, Dick, Mary Dalrymple, Ron Kaplan, Tracy King, John Maxwell & Paula Newman. 2011. XLE documentation. https://ling.sprachwiss.uni-konstanz.de/pages/xle/doc/xle_toc.html.

Dalrymple, Mary (ed.). 1999. *Semantics and syntax in Lexical Functional Grammar: The resource logic approach*. Cambridge, MA: MIT Press. DOI: https://doi.org/10.7551/mitpress/6169.001.0001.

Dalrymple, Mary, John J. Lowe & Louise Mycock. 2019. *The Oxford reference guide to Lexical Functional Grammar*. Oxford: Oxford University Press. DOI: https://doi.org/10.1093/oso/9780198733300.001.0001.

Dalrymple, Mary, Agnieszka Patejuk & Mark-Matthias Zymla. 2020. XLE+Glue – A new tool for integrating semantic analysis in XLE. In Miriam Butt & Tracy Holloway King (eds.), *The proceedings of the LFG'20 conference*, 89–108. Stanford, CA: CSLI Publications. https://web.stanford.edu/group/cslipublications/cslipublications/LFG/LFG-2020/.

Davidson, Donald. 1967. The logical form of action sentences. In Rescher (1967) 81–95.

Findlay, Jamie Y. & Dag T. T. Haug. 2022. Managing scope ambiguities in Glue via multistage proving. In Miriam Butt, Jamie Y. Findlay & Ida Toivonen (eds.), *The proceedings of the LFG'22 conference*, 144–163. Stanford, CA: CSLI Publications. https://ojs.ub.uni-konstanz.de/lfg/index.php/main/issue/view/1.

Kokkonidis, Miltiadis. 2008. First-order Glue. *Journal of Logic, Language and Information* 17(1). 43–68. DOI: https://doi.org/10.1007/s10849-006-9031-0.

Meßmer, Moritz & Mark-Matthias Zymla. 2018. The glue semantics workbench: A modular toolkit for exploring linear logic and glue semantics. In Miriam Butt & Tracy Holloway King (eds.), *The proceedings of the LFG'18 conference*, 268–282. Stanford, CA: CSLI Publications. https://web.stanford.edu/group/cslipublications/cslipublications/LFG/LFG-2018/.

Park, Sang-Hee. 2019. *Gapping: A constraint-based syntax-semantics interface*. Ph.D. dissertation, State University of New York at Buffalo.

Parsons, Terence. 1990. *Events in the semantics of English: A study in subatomic semantics*. Cambridge, MA: MIT Press.

Partee, Barbara H. & Mats Rooth. 1983. Generalized conjunction and type ambigu-

ity. In Rainer Bäuerle, Christoph Schwarze & Arnim von Stechow (eds.), *Meaning, use and interpretation of language*, 361–383. Berlin: Walter de Gruyter. DOI: https://doi.org/10.1515/9783110852820.361.

Patejuk, Agnieszka & Adam Przepiórkowski. 2017. Filling the gap. In Miriam Butt & Tracy Holloway King (eds.), *The proceedings of the LFG'17 conference*, 327–347. Stanford, CA: CSLI Publications. http://web.stanford.edu/group/cslipublications/cslipublications/LFG/LFG-2017/.

Rescher, Nicholas (ed.). 1967. *The logic of decision and action*. Pittsburgh, PA: University of Pittsburgh Press.

Ross, John. 1970. Gapping and the order of constituents. In Manfred Bierwisch & Karl Erich Heidolph (eds.), *Progress in linguistics*, The Hague: Mouton.

Sag, Ivan A. 1976. *Deletion and logical form*. Cambridge, MA. Ph.D. dissertation, Massachusetts Institute of Technology.