Mapping Theory and the anatomy of a verbal lexical entry

Jamie Y. Findlay
University of Oxford

Proceedings of the LFG'20 Conference

On-Line

Miriam Butt, Ida Toivonen (Editors)

2020

CSLI Publications

pages 127–147

http://csli-publications.stanford.edu/LFG/2020

Keywords: mapping theory, argument structure, glue semantics, lexicon, templates

Findlay, Jamie Y. 2020. Mapping Theory and the anatomy of a verbal lexical entry. In Butt, Miriam, & Toivonen, Ida (Eds.), *Proceedings of the LFG'20 Conference, On-Line*, 127–147. Stanford, CA: CSLI Publications.

Abstract

This paper presents a new formal framework within which to implement (Lexical) Mapping Theory. It differs from previous accounts in that it is expressed in terms of vanilla LFG+Glue, rather than relying on any additional, bolted-on formal machinery.

1 Introduction

This paper proposes a new set of tools for describing the mappings between semantic arguments and their grammatical functions, the kind of linkings that are standardly handled in LFG by (Lexical) Mapping Theory (LMT).[1] Although I make some specific theoretical assumptions here – both for the sake of concreteness, and as a contribution to one particular strand of research – it is worth highlighting that the tools I describe in this paper are compatible with a number of different versions of LMT. As such, these proposals should be of interest even to those coming from quite different theoretical backgrounds.

The theoretical assumptions I make here are of two types. Firstly, the underlying theory of LMT I will be assuming is that of Kibort (2001, 2007, 2014), with some adjustments as described in Findlay (2016). Secondly, I will follow the architectural assumptions of a strand of research originating with Asudeh & Giorgolo (2012), namely that LFG does not require a dedicated level of a(rgument)-structure, since the same results can be achieved by using Glue Semantics and a more articulated s(emantic)-structure (on which see e.g. Asudeh et al. 2014, Findlay 2016, Lowe 2014, 2015, Lovestrand 2018). That research also emphasises the use of *templates* (Dalrymple et al. 2004, Crouch et al. 2017), labelled chunks of functional description, as a means of modularising the lexicon and capturing generalisations. The present work shares this focus, whilst also highlighting the role of templates in presenting complex formalism in a more user-friendly format.

The main goal of this paper is to describe a set tools which allows LMT to be implemented using only the vanilla LFG+Glue formalism, with no additional formal machinery required. Although the modular architecture of LFG allows for different levels of representation to have their own formal properties, I believe that

[†]This work has benefitted enormously from conversations over the years with Ash Asudeh, Miriam Butt, Anna Kibort, Joey Lovestrand, and John Lowe, to whom I wish to express my thanks. They won't agree with all I have to say here, however, and of course bear no responsibility for any errors. I would also like to thank the two reviewers for their detailed comments on an earlier draft: the present paper has been incalculably improved as a result.

[1]There are various names for the theory of the mapping between arguments and grammatical functions in the LFG literature: Lexical Mapping Theory is perhaps the most widespread, although several scholars have pointed out the problems with seeing the theory as applying purely in the lexicon (e.g. Butt 1995, Alsina 1996), and so often the 'Lexical' is dropped, giving us 'Mapping Theory' *tout court* (as in e.g. Kibort & Maling 2015). Other names include Functional Mapping Theory (e.g. Alsina 1996) and Linking Theory (e.g. Butt et al. 1997). In this paper I use 'LMT' as a cover term, without taking a particular position on what the theory ought ultimately to be called.

we should prefer sparser theories, all things being equal, and so if we can do without these extra mechanisms, we should. What is more, if new objects or operations are introduced into the framework, their formal properties should be rigorously defined, and this has not always been the case with LMT proposals.

In Section 2, I discuss some preliminaries, about LMT and about the use of templates. Sections 3 and 4 give the formal details of my proposal. Section 5 shows how this can be used to represent Kibort-LMT, and then Section 6 examines argument alternations using that theory. Section 7 concludes.

2 Background

2.1 Three facets of LMT

In this section, I will discuss what I believe are the three basic components of LMT, and, in so doing, offer some concrete points of departure for my own proposals.

2.1.1 Linking

In broadly theory-neutral terms, LMT is a theory of the linking between semantic arguments and grammatical functions. But already at this very abstract level there are technical questions: how should we actually implement this linking? Most work in LMT (e.g. Bresnan & Kanerva 1989), including the most fully developed modern implementation, that of Kibort (2001, 2007, *et seq.*), leaves the mapping relation totally unanalysed, and says nothing about how it is to be integrated into the LFG architecture. This is clearly problematic from the point of view of formalisation. Butt et al. (1997) were the first to take this problem seriously, proposing that linking is codescription – specifically, a set of constraints which describe the correspondence between a-structure and f-structure. Although Asudeh & Giorgolo (2012) highlight a number of problems with the architecture proposed by Butt et al. (1997) (see Findlay 2016: 303–309 for discussion), they also agree that linking should be handled by codescription, and this is the approach I will take here as well. As mentioned earlier, I assume that a-structure has been replaced by a connected s-structure, and so the mapping equations will describe the relation between f-structure and s-structure, via the σ-projection. This means that the functional description for a given analysis will contain equations like (1), which says that the AGENT argument of a predicate is associated with the SUBJ GF:

(1) $(\uparrow \textsc{subj})_\sigma = (\uparrow_\sigma \textsc{agent})$

Knowing what the nature of the relation between arguments and GFs is is one fundamental component of LMT. But it doesn't address the question of how we decide which GFs are linked with which arguments. There are two aspects to this. Firstly, we need an account of the *possible* GFs an argument can be associated with, incorporating a degree of flexibility to allow for argument alternations; secondly, we need a way of resolving this indeterminacy so that we arrive at the final

mapping.

2.1.2 Mapping possibilities

The first of these challenges is generally met in LMT by associating each argument with a pair of GFs, via a feature decomposition which breaks GFs down into purportedly natural classes. The standard breakdown of GFs is as follows (Bresnan & Kanerva 1989: 24f.):

(2)

	$-r$	$+r$
$-o$	SUBJ	OBL_θ
$+o$	OBJ	OBJ_θ

Each argument can then be associated with a single feature, say $[-o]$, which allows it to be realised as one of the two compatible GFs, in this case SUBJ or OBL_θ. This is often done on the basis of intrinsic classifications, e.g. Agent-like arguments are intrinsically $[-o]$, Patient-like arguments are $[-r]$, etc. There is also scope for cross-linguistic variation in how different kinds of arguments are classified.

There are certain unanswered questions about the exact status of these features (see Findlay 2016: 298f. for some discussion), and about whether they have any real explanatory value (does saying an argument is $[+o]$ say anything more than that it is realised as an OBJ or OBJ_θ?). As Findlay (2016: 299) concludes, however, all that is essential is just that each argument should be associated with a pair of GFs, and we can achieve this in a more explicit way by simply defining four abbreviations:

(3)
a. $\text{MINUSO} \equiv \{\text{SUBJ} \mid \text{OBL}_\theta\}$
b. $\text{PLUSO} \equiv \{\text{OBJ} \mid \text{OBJ}_\theta\}$
c. $\text{MINUSR} \equiv \{\text{SUBJ} \mid \text{OBJ}\}$
d. $\text{PLUSR} \equiv \{\text{OBJ}_\theta \mid \text{OBL}_\theta\}$

Of course, insofar as the feature decompositions really do capture natural classes, this arbitrary listing is unsatisfactory. However, as a reviewer laments, the features $[\pm o/r]$ in fact describe an unnatural class (no alternation involves the two $[+r]$ GFs, OBJ_θ and OBL_θ) and also fail to describe a natural class, that of *terms* or *direct* GFs, *viz.* SUBJ, OBJ, and OBJ_θ (Alsina 1996: 29, fn. 9). Other feature decompositions are certainly possible (Alsina 1996: 19f. uses the features $[\pm\text{obl}/\text{subj}]$, for instance), and may be more satisfactory from a descriptive point of view, but using explicit disjunctions is the more conservative approach: at worst we are missing a more profound explanation for our generalisations (though we can still express such generalisations), but this is preferable to introducing new entities to our formal ontology which may not be well motivated. And although there may be issues with it, the $[\pm o/r]$ decomposition, and thus the groupings given in (3), is still very much the orthodox view (it is the one presented in textbooks/handbooks such as Bresnan et al. 2016: Ch. 14, Dalrymple et al. 2019: Ch. 9, and Börjars et al. 2019: Ch. 8, for instance), and so it is the system I will work with here.

2.1.3 Mapping resolution

Once we have associated each of a predicate's arguments with a pair of GFs, the final step in mapping is to resolve this indeterminacy and select a single GF for each. How is this done?

All versions of LMT appeal to some kind of ranking of a-structure arguments, along with a ranking of GFs. Often these take the following forms:

(4) **Thematic hierarchy** (Bresnan & Kanerva 1989: 23)
Agent $>$ Beneficiary $>$ Recipient/Experiencer
$>$ Instrument $>$ Theme/Patient $>$ Location

(5) **Grammatical function hierarchy** (Bresnan & Zaenen 1990: 49)
SUBJ $>$ OBJ, OBL_θ $>$ OBJ_θ

Mapping is then a question of correctly harmonising these two hierarchies. In classical LMT (Bresnan & Kanerva 1989, Bresnan & Zaenen 1990), arguments in a-structure are ordered left-to-right according to their position in the thematic hierarchy (with the leftmost argument having the highest thematic role). Various *Mapping Principles* then assign mappings according to certain ranking properties: if the leftmost argument is $[-o]$, it is mapped to SUBJ, otherwise a $[-r]$ argument is; of the remaining arguments, the highest thematic roles are mapped to the lowest GFs compatible with their intrinsic features.[2]

The use of a thematic hierarchy for these purposes is hugely problematic, however. As has been frequently discussed, a satisfactory list of thematic roles has never been given (Dowty 1991, Davis 2011), and even when a set of roles is agreed on, different phenomena seem to call for different orderings in the hierarchy (Rappaport Hovav & Levin 2007), all of which renders the explanatory power of a purportedly universal thematic hierarchy highly suspect. For this reason, Kibort (2007) advocates a separation of a-structure argument positions from thematic roles (a return to the earlier LFG position of e.g. Bresnan 1982), with a-structures instead containing *sui generis* argument positions, drawn from a universally available set of ordered arguments, each intrinsically associated with one of the LMT mapping features (arg_1 is normally associated with $[-o]$, but $[-r]$ is used for unaccusatives):

(6)

$\langle$	arg_1	arg_2	arg_3	arg_4	$\ldots$	arg_n	$\rangle$
	$[-o]/[-r]$	$[-r]$	$[+o]$	$[-o]$		$[-o]$	

This allows Kibort to reduce the classical LMT Mapping Principles to a single principle: the highest arguments are mapped onto the *highest* compatible GFs.

However, both the classical and the modern approaches to LMT suffer from

[2]Classical LMT also requires two well-formedness principles to rule out some unwanted mappings that would otherwise be licensed by the Mapping Principles. Firstly, the *Subject Condition* ensures that there must be a SUBJ. Secondly, *Function-Argument Biuniqueness* ensures that no two arguments map to the same GF, and vice versa. Kibort's version of LMT makes the Subject Condition redundant, and the proposals in this paper also make Function-Argument Biuniqueness unnecessary.

what seems to me a critical flaw: the actual application of the Mapping Principle(s) happens 'off stage'. There is no explanation of how the harmonisation between the two hierarchies (whichever two are used) might be achieved using the existing LFG machinery, and in fact no formal details of the implementation are usually given at all. Once again, Butt et al. (1997) are the exception here, since they give an explicit algorithm for resolving the final mapping. But their approach uses a separate system which is wholly outside of the LFG architecture. It seems to me something of an embarrassment for LMT that carrying out its very *raison d'être*, determining the linking between semantic arguments and GFs, must ultimately be outsourced. It would be far better if LFG were able to do the heavy lifting 'in house', and one of the goals of this paper is to show that this is indeed possible.

2.2 Modularisation of the lexicon

The other goal is to show that such a theory can be developed in a way which contributes to the ongoing project of modularising the lexicon. This refers particularly to the use of *templates* (Dalrymple et al. 2004, Crouch et al. 2017) to factor out information and allow generalisations to be expressed.

A template is just an abbreviation for a piece of functional description. It can be used to label annotations which often appear together, for example. (7) is a very simple lexical entry for the third-person singular verb form *protects* in English:

(7) protects V $(\uparrow \text{PRED}) = \text{'protect'}$
$(\uparrow \text{TENSE}) = \text{PRESENT}$
$(\uparrow \text{SUBJ NUM}) = \text{SG}$
$(\uparrow \text{SUBJ PERS}) = 3$

Since any third-person singular verb form in English will also contain the final two lines, we can bundle them together into a template called 3SG:

(8) $3\text{SG} :=$
$(\uparrow \text{SUBJ NUM}) = \text{SG}$
$(\uparrow \text{SUBJ PERS}) = 3$

Then we can rewrite the lexical entry using this template:

(9) protects V $(\uparrow \text{PRED}) = \text{'protect'}$
$(\uparrow \text{TENSE}) = \text{PRESENT}$
@3SG

The **@** symbol is prefixed to a template name to 'call' it. Calling a template just replaces it with its contents. The lexical entries in (7) and (9) are therefore extensionally equivalent – they contain the same functional description – it's just that the latter expresses a generalisation rather more transparently. Templates can also be made more flexible by parametrising them. (10) gives a very simple example:

(10) $\textsc{Tense}(X) :=$
$(\uparrow \textsc{tense}) = X$

This template can be called with an argument, and that argument will appear as the value of ↑'s TENSE feature. We can thus further modify our running example:

(11) protects V $(\uparrow \textsc{pred}) =$ 'protect'
@TENSE(PRESENT)
@3SG

There are a number of advantages, both practical and theoretical, to using templates to break down lexical entries in this way. Firstly, they make grammar engineering much more robust, since if something needs to be changed in a grammar a single template definition can be modified, rather than having to go in and change every relevant lexical entry individually, which would inevitably lead to errors. Secondly, they can make analyses more readable and make theoretical tools more user friendly, by concealing formal 'gore' but leaving the theoretically interesting claims exposed. This is akin to how modern programming languages abstract away from the machine code which ultimately implements a program. This kind of motivation has been present in LFG since the start – for example, in the use of the more readable ↑ to abbreviate the clunkier $\phi(\mathcal{M}(*))$. We can see a thoroughgoing templatic approach as offering an interface to the LFG formalism which is easier for the theorist to work with. Lastly, templates can make important empirical generalisations easier to see and easier to represent. By searching for ways to abbreviate lexical entries, we can be led to notice where they share information and where they differ from one another – and where they do differ, if they do so only in limited ways (where a template could be parametrised) or more profoundly. In this way, templates can also help mark the distinction between formalism and theory: while the LFG formalism itself places relatively weak limitations on what can be expressed, our choice of basic templates can impose strong limits.

The rest of this paper will present an analysis of verbal lexical entries whereby they can be broken down into the following parts:

(12) verb V **[core information]**
[valency information]
[argument alternations]
[other information]

The first three parts of the functional description are the focus of this paper, since they relate to mapping. Each verb must contain some core information about the relation it expresses. What exactly this encompasses is the focus of Section 3. It must also contain information about its valency: this part of the entry identifies the arguments of a predicate, what roles they play in the eventuality described by the predicate, and how they are realised syntactically; Section 4 details how this is accomplished. In addition to a verb's basic/underived valency frame, there can

also be extra information realising various morphosyntactic argument alternations, like the passive. Section 6 addresses this component. Finally, there will be other information – about agreement, for example, or relating to idiosyncratic features of the particular verb. Here too it is to be hoped that sub-regularities can be found which can then be factored out into templates (as with the example of 3SG above), but this will not be my focus here.

3 Core information

There are two sides to the core information section of a verbal lexical entry. On the grammatical side, the entry must provide a PRED value for f-structure, and a REL value for s-structure. On the meaning side, it must provide a meaning constructor expressing what kind of eventuality the verb denotes.

The current status of PRED and REL is far from settled: many if not all of the important functions of PRED have been taken over by Glue Semantics (Andrews 2008), and REL really has no substantive role in the theory (Lovestrand 2018: 169ff.; although see Lowe 2014), but I include both for consistency with other work which makes use of them. Assuming that PRED and REL always have the same value (Lovestrand 2018: 170), the grammatical side of things can easily be expressed in a template PRED-REL:

(13) $\textsc{Pred-Rel}(X) :=$
$(\uparrow \textsc{pred}) = \text{`}X\text{'}$
$(\uparrow_\sigma \textsc{rel}) = X$

We can then combine this with the meaning constructor in a template VERB-LEXEME:[3]

(14) $\textsc{Verb-Lexeme}(\textit{pred-rel}, \textit{meaning}) :=$
$@\textsc{Pred-Rel}(\textit{pred-rel})$
$\lambda e.\textit{meaning}(e) : (\uparrow_\sigma \textsc{event}) \multimap \uparrow_\sigma$

Let us take the verb *give* as our running example going forward. Its core information would be expressed as in (15), which is equivalent to (16):

(15) give V $@\textsc{Verb-Lexeme}(\text{give}, \mathbf{give})$

(16) give V $(\uparrow \textsc{pred}) = \text{`give'}$
$(\uparrow_\sigma \textsc{rel}) = \text{give}$
$\lambda e.\mathbf{give}(e) : (\uparrow_\sigma \textsc{event}) \multimap \uparrow_\sigma$

[3] I am making a number of simplifying assumptions here. Firstly, there are well-known problems with treating verbal meanings as being of type $\langle v, t\rangle$ relating to scopal interactions (or the lack thereof), and we should instead use a higher type $\langle\langle v, t\rangle, t\rangle$, as advocated by Champollion (2015). But for the sake of simplicity I stick to the lower type here, since this is not a focus of the present paper. Secondly, the meaning constructor assumes that the verb denotes a predicate of *events*, and ignores *states* – stative verbs will require a subtly different treatment, and this distinction could easily be factored out by using further templates.

4 Valency information

The valency information of a verb has five parts, which will be presented in turn:

1. A meaning constructor which connects the arguments to the event described by the verb.
2. A set of existential constraints, requiring the presence of these arguments.
3. An expression identifying the default logical subject.
4. A set of DEFAULT-MAPPING templates, determining the default linking between the arguments and their GFs.
5. A set of PREFERRED-MAPPING templates, which implement LMT's Mapping Principle(s).

4.1 Valency meaning constructor

The valency meaning constructor consumes a verbal meaning and returns a new meaning constructor which consumes the arguments of the verb. In other words, it raises the type of the verb from the simple $\langle v, t\rangle$ to a higher type with more dependents: $\langle \tau_1, \ldots, \tau_n, \langle v, t\rangle\rangle$, where τ_1 is the type of the first argument, and τ_n the type of the last argument. For our running example, this will be (17):

(17) $\lambda P \lambda x \lambda y \lambda z \lambda e.P(e) \wedge \mathbf{agent}(e, x) \wedge \mathbf{theme}(e, y) \wedge \mathbf{beneficiary}(e, z) :$
$[(\uparrow_\sigma \text{EVENT}) \multimap \uparrow_\sigma] \multimap$
$(\uparrow_\sigma \text{AGENT}) \multimap (\uparrow_\sigma \text{THEME}) \multimap (\uparrow_\sigma \text{BEN}) \multimap (\uparrow_\sigma \text{EVENT}) \multimap \uparrow_\sigma$

I use thematic role names to label s-structure arguments purely for readability; this choice has no theoretical significance, and I continue to assume, following e.g. Kibort (2007) and Asudeh & Giorgolo (2012), that thematic role information is best left out of the grammar itself and relegated to the meaning language (see also Findlay 2016: 314). The choice of s-structure argument labels is arbitrary, and we could as well have used ARG1, ARG2, etc. – the reason I have not done so here is in order to avoid confusion with the argument positions in Kibort's valency frame. In Findlay (2016), I took the two to be equivalent, but this is an unnecessary restriction to impose on the formalism, and one which weds it too closely to one particular theory. We will see below how that information can instead be encoded using local names if we wish to implement Kibort's theory.

4.2 Existential constraints

Once again in contrast to Findlay (2016: 320, fn. 19), I do not assume that being mentioned in a meaning constructor is sufficient for an attribute to appear at s-structure. Instead, the lexical entry for a verb also includes an existential constraint for each of the arguments mentioned in the valency meaning constructor, requiring its presence. A constraint like '$(\uparrow_\sigma \text{AGENT})$' requires there to be a positive specification somewhere in the functional description which provides the contents of the AGENT argument position at s-structure. The normal way for this to

happen is for the argument to be linked to a GF – then a lexically-specified REL value can be passed on. These existential constraints therefore effectively require that the arguments they mention participate in mapping, unless some argument suppressing operation can be appealed to.

4.3 Specification of logical subject

Many mapping theories appeal to a privileged, 'most prominent', argument structure position, sometimes called the *logical subject*, and also denoted $\hat{\theta}$, for 'highest thematic argument' (Bresnan & Kanerva 1989, Alsina 1996: 36f.). By analogy with Falk's (2006) $\widehat{\text{GF}}$, I propose to call this position $\widehat{\text{ARG}}$. Part of the valency information encoded in a verbal lexical entry includes which of its arguments is, by default, the logical subject:

(18) $\quad$ DEFAULT-$\widehat{\text{ARG}}(arg) :=$
$\quad\quad \left\{(\uparrow_\sigma \widehat{\text{ARG}}) = (\uparrow_\sigma arg) \mid (\uparrow_\sigma \widehat{\text{ARG}})\right\}$

This uses the basic approach to defaults described by Dalrymple et al. (2004: 205f.): the left hand disjunct must be true unless something else provides the appropriate information, in which case the right-hand side can be true instead – in other words, the left-hand disjunct will hold by default. The reason we cannot specify the logical subject once and for all is that certain processes, like causativisation, can add a new logical subject, which therefore overrides the default.[4]

In most formulations of LMT, the choice of logical subject follows from other properties – usually from the ranking of arguments according to the thematic hierarchy. Here we merely stipulate it, which may seem unsatisfactory by comparison. However, recall that there is no agreed-upon/adequate thematic hierarchy, and so appeals to such a mechanism are in fact spurious. Once again, I take the conservative view that encoding this information directly in the lexical entry is not a bad thing for the time being. It may well be that this information can be made to follow from other properties, especially if the contents of s-structure is further developed, for example to include proto-role information. Until then, the direct lexical specification can be taken as a stand-in for whatever the proper mechanism is.

With that said, we can still capture certain generalisations by using templates, so that what counts as the logical subject is not precisely a matter of *lexical* stipulation; rather, it will be a property shared by certain kinds of verbs that express the same kinds of thematic roles. For example, our running example of *give* is a verb which takes an Agent, a Theme, and a Beneficiary argument, and whichever argument corresponds to the Agent will be the default $\widehat{\text{ARG}}$. *Give* will share the valency information we have described so far with other verbs like it, a fact we can capture in a template called by all such verbs:

[4] Although many complex predicates can be handled straightforwardly, along the same lines as Lowe (2015), recursive causatives will cause problems, and this is an area that needs further work.

(19) AGENT-THEME-BENEF-VERB(*arg1*, *arg2*, *arg3*) :=
$\lambda P \lambda x \lambda y \lambda z \lambda e.P(e) \wedge \mathbf{agent}(e, x) \wedge \mathbf{theme}(e, y) \wedge \mathbf{beneficiary}(e, z)$:
$[(\uparrow_\sigma \text{EVENT}) \multimap \uparrow_\sigma] \multimap$
$(\uparrow_\sigma \textit{arg1}) \multimap (\uparrow_\sigma \textit{arg2}) \multimap (\uparrow_\sigma \textit{arg3}) \multimap (\uparrow_\sigma \text{EVENT}) \multimap \uparrow_\sigma$

$(\uparrow_\sigma \textit{arg1}) \wedge (\uparrow_\sigma \textit{arg2}) \wedge (\uparrow_\sigma \textit{arg3})$

@DEFAULT-$\widehat{\text{ARG}}$(*arg1*)

4.4 Default mapping

In Section 2.1, we identified two tasks for a mapping theory: to associate an argument with a pair of GFs, and then to decide between those GFs in different situations. In the present proposal, I divide up the task slightly differently. Instead of describing a set of possible GFs and then deciding between them, we will first describe, for each argument, a *default* mapping to a GF, and then we will describe a *preferred* mapping. This reflects the two-dimensional information present in many LMT a-structure representations, owing to the ranking of the arguments alongside their association with a pair of GFs.

For example, because Kibort's Mapping Principle links the highest arg positions to the highest GFs, each position below arg_1 in Kibort's theory is essentially in competition with some higher arg position:[5]

- arg_2 would prefer to be a SUBJ, the highest ranked $[-r]$ GF, but is generally blocked from doing so by arg_1, and so defaults to OBJ.
- arg_3 would prefer to be an OBJ, the highest ranked $[+o]$ GF, but is generally blocked from doing so by arg_2, and so defaults to OBJ_θ.
- arg_4 would prefer to be a SUBJ, the highest ranked $[-o]$ GF, but is generally blocked from doing so by arg_1, and so defaults to OBL_θ.

The next section explains how the preferred mapping is handled; in this section I show how we can achieve the default effect. In order to do this, we make use of a two-member disjunction, as we did when describing the default logical subject. The contents of this disjunction are given by the template DEFAULT-MAPPING, which takes three parameters: the default GF, the argument name, and a disjunction identifying the set of *disallowed* GFs (e.g. if the argument is assigned to the $[-r]$/MINUSR pair of GFs in the Kibort mapping theory, this parameter will be set to PLUSR):

(20) DEFAULT-MAPPING(*default-GF*, *arg*, *disallowed-GFs*) :=
$$\left\{ \textbf{@}\text{MAP}(\textit{default-GF}, \textit{arg}) \;\middle|\; \begin{array}{l} \neg\textbf{@}\text{MAP}(\textit{default-GF}, \textit{arg}) \\ \neg\textbf{@}\text{MAP}(\textit{disallowed-GFs}, \textit{arg}) \end{array} \right\}$$

[5] I follow Findlay (2016: 317f.) in assuming that only the first four arguments in Kibort's valency frame participate in mapping and in argument alternations, the others being treated as 'derived arguments' (Needham & Toivonen 2011) and added via various lexical or syntactic processes.

Let us look at an example to see how this works. First of all, though, we need to define the MAP template. It maps its first parameter, a grammatical function *GF*, to its second parameter, an s-structure argument *arg*.

(21) $\textsc{Map}(GF, arg) \coloneqq$
$(\uparrow GF)_\sigma = (\uparrow_\sigma arg)$

Consider the case of a Kibort arg_2, i.e. a $[-r]$ argument, and assume it is associated with a THEME argument at s-structure. Then we would call the template like this:

(22) **@**DEFAULT-MAPPING(OBJ, THEME, PLUSR)

When expanded, this gives us the following:

(23) $\left\{ \textbf{@}\textsc{Map}(\text{OBJ}, \text{THEME}) \;\middle|\; \begin{array}{l} \neg\textbf{@}\textsc{Map}(\text{OBJ}, \text{THEME}) \\ \neg\textbf{@}\textsc{Map}(\text{PLUSR}, \text{THEME}) \end{array} \right\}$

This disjunction requires either that OBJ is mapped to the THEME argument, or alternatively that neither OBJ nor one of the $[+r]$ GFs, OBJ_θ or OBL_θ, is mapped to it – which leaves only one option for mapping: SUBJ. Crucially, though, this non-canonical mapping is only indirectly licensed: it is given in negative rather than positive terms. This is what induces the 'default' behaviour: in the absence of further information, the first disjunct must be true, since the existential constraint introduced in the verb's valency template requires that there be *some* positive specification of the mapping for the THEME argument. If we try to make the second disjunct true, we end up with a collection of negative constraints but no positive ones, and so the relevant existential constraint is not satisfied. Thus, without further specification – from an argument alternation like passive, for example – the default mapping prevails.

The DEFAULT-MAPPING template gives us a general tool for associating an argument with a pair of GFs, where one of them is identified as a default – i.e. the GF to which the argument will normally be linked, all things being equal. This can be used to implement any number of specific theories about the actual connections between arguments and GFs. In Section 5, it will be one of the tools we use to implement Kibort's version of LMT. For now, though, we need one additional tool: a means of capturing the competition between arguments for GFs.

4.5 Preferred mapping

Of course, arguments do not always surface as their default GFs. There is usually another GF which an argument will surface as if it is given the opportunity to – for example, because another argument has been suppressed or had its mapping possibilities altered by some morphosyntactic process. We call this the argument's preferred GF. For example, as I described earlier, a Kibort arg_2 will surface as a SUBJ if nothing else has a better claim to the SUBJ position, i.e. if arg_1 does not take it (because there is no arg_1 or because it is suppressed by passive, etc.). To capture

this fact, we define a template PREFERRED-MAPPING, intended to be used alongside a call of the DEFAULT-MAPPING template which involves the same argument. The PREFERRED-MAPPING template takes two parameters: the preferred GF of an argument, and the name of that argument.

(24) $\textsc{Preferred-Mapping}(GF, arg) :=$
$$\left\{ (\uparrow GF)_\sigma = (\uparrow_\sigma arg) \;\middle|\; \begin{array}{c} (\uparrow GF) \\ (\uparrow GF)_\sigma \neq (\uparrow_\sigma arg) \end{array} \;\middle|\; @\textsc{NoMap}(\text{arg}) \right\}$$

To see how this works, let us again look at the example of an arg_2 THEME in Kibort's LMT. The appropriate template call for this situation is given in (25):

(25) @PREFERRED-MAPPING(SUBJ, THEME)

When expanded, this gives the following:

(26) $$\left\{ (\uparrow \text{SUBJ})_\sigma = (\uparrow_\sigma \text{THEME}) \;\middle|\; \begin{array}{c} (\uparrow \text{SUBJ}) \\ (\uparrow \text{SUBJ})_\sigma \neq (\uparrow_\sigma \text{THEME}) \end{array} \;\middle|\; @\textsc{NoMap}(\text{THEME}) \right\}$$

This disjunction offers us three possibilities:

1. The THEME argument is mapped to SUBJ (its preferred GF).
2. Something else is mapped to SUBJ (i.e. there is a SUBJ but it isn't the THEME argument).
3. The THEME argument is not mapped to any GF.

The third option is realised by a template NOMAP, defined in (27):

(27) $\textsc{NoMap}(arg) :=$
$(\uparrow_\sigma arg)_{\sigma^{-1}} = \varnothing$

It describes a situation where no GF is linked to the argument in question (by stating that the inverse of the σ-projection, taking us from s-structure to f-structure, is empty when applied to it). This will only be relevant in cases of argument suppression, since otherwise the existential constraint on the argument in question introduced in the verb's valency template will require that *something* maps to it. We will discuss argument suppression in Section 6.2, but for now we can safely ignore this option, meaning our choice is between the first two disjuncts.

In the canonical mapping for a transitive verb like *kill*, which has an arg_1 and an arg_2 in Kibort-LMT terms, the arg_1 will map to SUBJ. This means that the first disjunct in (26) cannot be true, since the σ-projection is a function, and so it cannot link the same f-structure element to multiple s-structure elements.[6] That means that in the canonical/default situation, the second disjunct must be true. All this does is add yet another negative constraint to the mapping possibilities for this

[6]Note that this makes at least one component of Function-Argument Biuniqueness otiose. The other part, prohibiting multiple GFs from mapping to the same argument, is not necessary either, provided that the grammar simply never makes such possibilities available: in the present proposal, the only mapping possibilities are those explicitly introduced in the mapping templates.

argument, which once again means it is the first disjunct of the relevant DEFAULT-MAPPING template which must hold, since that remains the only positive mapping specification available for the argument in question.

The existential constraint introduced in the second disjunct ensures that the default mapping only obtains if this argument's preferred GF is not available. If nothing else maps to SUBJ, then the THEME argument should do so – that's what it means for it to be the argument's preferred GF. If that does not happen, then there will be no SUBJ, which means the second disjunct here will be rendered false because the existential constraint will not be satisfied.

Overall, the PREFERRED-MAPPING template gives us a means of associating an argument with a preferred GF – one which it will map to if given the opportunity, i.e. if nothing else is required to map to it in preference. Combined with the DEFAULT-MAPPING template, this allows us to simulate the effects of a hierarchy of arguments/GFs, commonly appealed to in LMT.

5 Kibort-LMT

In this section, I will show how we can implement (a version of) Kibort's LMT, using the tools developed in the previous section. Each of the positions in Kibort's valency frame provides a default and a preferred GF, which we now have the means to represent. What is more, each position can be referred to by other processes, e.g. locative inversion adds a $[+o]$ specification to arg_1 specifically, and so we also need a means of labelling each argument position. We achieve this using *local names* (Crouch et al. 2017). A local name, indicated by a prefixed $\%$, is essentially a variable name: it allows for a particular entity to have a name which can be used to refer to it within the same local description – here this will mean the same lexical entry. This enables other templates, e.g. those encoding morphosyntactic argument-manipulating operations, to refer to specific argument positions by name.

We define a template for each of the arg positions. These templates take a single parameter: the name of an s-structure argument. For the first position in Kibort's valency frame, arg_1, we include only a default mapping, not a preferred one, since its default is already the highest available GF. Arg_1 can have two specifications. Normally it will be $[-o]$:

(28) ARG1(*arg*) :=
@DEFAULT-MAPPING(SUBJ, *arg*, PLUSO)
arg = %arg1

For unaccusatives, however, it is $[-r]$:

(29) ARG1-UNACCUSATIVE(*arg*) :=
@DEFAULT-MAPPING(SUBJ, *arg*, PLUSR)
arg = %arg1

I defer discussion of arg_2 momentarily, since it involves a small additional com-

plexity. The other two arg positions are straightforward, however:[7]

(30) ARG3(arg) :=
@DEFAULT-MAPPING(OBJ$_\theta$, arg, MINUSO)
@PREFERRED-MAPPING(OBJ, arg)
arg = %arg3

(31) ARG4(arg) :=
@DEFAULT-MAPPING(OBL$_\theta$, arg, PLUSO)
@PREFERRED-MAPPING(SUBJ, arg)
arg = %arg4

The arg_2 position requires one extra constraint, which is boxed in (32):

(32) ARG2(arg) :=
@DEFAULT-MAPPING(OBJ, arg, PLUSR)
@PREFERRED-MAPPING(SUBJ, arg)
$\boxed{(\uparrow \text{SUBJ})_\sigma \neq (\uparrow_\sigma arg) \Rightarrow (\uparrow \text{SUBJ})_\sigma = (\uparrow_\sigma \widehat{\text{ARG}})}$
arg = %arg2

This conditional constraint (Bresnan et al. 2016: 60f.) says that if this argument is not mapped to SUBJ, then the argument that is must be the $\widehat{\text{ARG}}$. Since the logical subject is the highest ranked argument in a-structure, this means that the only time the arg_2 won't map to SUBJ is when the (higher ranked) arg_1 does. This gives arg_2 priority for the subject slot over arg_4, correctly capturing their hierarchical ranking.

Let us illustrate how these templates can be used via our running example. *Give* participates in the dative shift alternation, which means that in Kibort-LMT terms it has two possible argument structures and corresponding alignments of semantic arguments with arg positions:

(33) a. $\text{Odo}_{\text{arg}_1}$ gave a $\text{gift}_{\text{arg}_2}$ to $\text{Kira}_{\text{arg}_4}$.
$\langle$ arg_1 arg_2 arg_4 $\rangle$
$[-o]$ $[-r]$ $[-o]$

b. $\text{Odo}_{\text{arg}_1}$ gave $\text{Kira}_{\text{arg}_2}$ a $\text{gift}_{\text{arg}_3}$.
$\langle$ arg_1 arg_2 arg_3 $\rangle$
$[-o]$ $[-r]$ $[+o]$

Here the dative-shifted version (33b) completely replaces one argument (arg_4) with another (arg_3), and also realigns the participants – the Theme, *a gift*, is now an arg_3, and arg_2 represents the Beneficiary, *Kira*, when in (33a) it was arg_2 which corresponded to the Theme and arg_4 which represented the Beneficiary. In order to capture this in the current system, we give both options in a disjunction:[8]

[7] It is not altogether clear whether arg_3 ever actually participates in any (strictly morphosyntactic) alternations. For example, the Patient/Theme argument of the Chicheŵa instrumental applicative can never trigger the appearance of an object marker on the verb, even in the passive (Alsina & Mchombo 1993), which means that even when the active-voice object is promoted to subject, the Patient/Theme argument remains an OBJ$_\theta$, and does not occupy the now vacant OBJ position. If this pattern obtains generally, then it might be sensible to simply specify arg_3 concretely as an OBJ$_\theta$, and avoid the unnecessary complexity of the mapping templates.

[8] A reviewer comments that it would be better if these possibilities followed from some semantic

(34) EN-DATIVE-SHIFT-MAPPING(ag, th, ben) :=
@ARG1(ag)
$\left\{ \begin{array}{l|l} \textbf{@}\text{ARG2}(th) & \textbf{@}\text{ARG2}(ben) \\ \textbf{@}\text{ARG4}(ben) & \textbf{@}\text{ARG3}(th) \end{array} \right\}$

Our running example, *give*, now has the following lexical entry:[9]

(35) give V **@**VERB-LEXEME(give, **give**)
@AGENT-THEME-BENEF-VERB(AGENT, THEME, BEN)
@EN-DATIVE-SHIFT-MAPPING(AGENT,THEME,BEN)

6 Argument alternations and argument suppression

In this section, we will see how argument alternations, like the locative inversion, and argument suppressing operations, like the short (agentless) passive, can be handled in the present system. The first kind involve a straightforward translation of Kibort's theory into the present formalism. However, Kibort-LMT has little to say about true argument suppression, since it simply assumes that obliques are always optional, and does not consider the semantic implications. We therefore have a little more work to do in that area.

6.1 Argument alternations

In keeping with the monotonic approach of Kibort-LMT, argument alternations involve adding further positive feature specifications to certain argument positions. We can emulate this directly, using the abbreviations PLUSO and PLUSR along with the MAP template. Let us consider the familiar locative inversion in Chicheŵa to see how this works.

This alternation was one of the first given an analysis in LMT, by Bresnan & Kanerva (1989). It is illustrated in (36), taken from Bresnan & Kanerva (1989: 2):

(36) a. [Chi-tsîme]$_{\text{SUBJ}}$ chi-li [ku-mu-dzi]$_{\text{OBL}_{\text{LOC}}}$.
7-well *7*SUBJ-*be 17-3-village*
'The well is in the village.'

b. [Ku-mu-dzi]$_{\text{SUBJ}}$ ku-li [chi-tsîme]$_{\text{OBJ}}$.
17-3-village *17*SUBJ-*be 7-well*
'In the village is a well.'

These predicates' a-structures contain an arg_1 and an arg_4 in Kibort-LMT terms – hence in the uninverted construction, (36a), they surface as a SUBJ and an OBL_θ.

properties of the verb, rather than being lexically stipulated. Once again, they are not strcitly *lexically* stipulated, since the template in (34) will be called by all verbs of this class. And again, a more fully developed theory of s-structure, perhaps along the lines of Jackendoff (1990), might help here – for example, by giving us somewhere to encode lexical semantic properties such as aspectual class.

[9] It would likely be sensible to collapse the second and third templates into one macro-template for verbs of this type, but I leave them separate here for the sake of exposition.

Kibort (2007) models locative inversion as the adding of a $[+o]$ feature to a $[-r]$, unaccusative, arg_1, fully specifying it as an OBJ; this allows the arg_4 to take the vacant SUBJ position, giving us the arrangement in (36b). We can represent this process using the following template:

(37) LOCATIVE-INVERSION :=
@MAP(PLUSO, %arg1)

Because the template ARG1-UNACCUSATIVE gives its argument the local name %arg1, other pieces of functional description, like the template in (37), can refer to that argument without worrying about what the actual s-structure attribute name is (it might be THEME, it might be ARG1, or it might be something else altogether).

The effect of (37) is to give a positive specification to the Theme argument, stating that the GF that maps to it must be either OBJ or OBJ_θ (i.e. a $[+o]$ GF). This is now incompatible with the default mapping, which is SUBJ. The only other possibility permitted by the DEFAULT-MAPPING template called by ARG1-UNACCUSATIVE (see (29)) is for the argument to be linked to OBJ, although it does not provide this possibility directly. The template in (37) does, however, and so now the only positive mapping specification which can hold is that OBJ maps to the argument in question. Given this, the preferred GF of the Location argument, SUBJ, is now available, and so it is mapped to this argument. If it was not, then the constraints in the PREFERRED-MAPPING template called by ARG4 would not be satisifed, since the argument in question would not be linked to SUBJ (first disjunct is false), but nor would any other argument (second disjunct is also false).

The long passive in English can be given a similar analysis:[10]

(38) LONG-PASSIVE :=
@MAP(PLUSR, %arg1)

This restricts a regular, $[-o]$ arg_1 in Kibort-LMT terms so that it can only appear as a $[+r]$ GF, i.e. one of OBJ_θ or OBL_θ. This prohibits the default mapping, which is to SUBJ, and the only other possibility permitted by the relevant DEFAULT-MAPPING template is OBL_θ, hence this argument emerges as a prepositional *by*-phrase.

6.2 Argument suppression

Argument suppressing operations carry an extra challenge compared with argument alternations, since the argument being suppressed will have both grammatical and semantic dependencies which must be dealt with. On the grammatical side, we must find a way to satisfy the existential constraint which would normally ensure that the argument is mapped to a GF. On the semantic side, we must do something

[10] I follow Huddleston & Pullum (2002: 1428) in referring to the version of the passive where the active-voice subject is expressed as a *by*-phrase as the 'long' passive, and the version where it is unexpressed as the 'short' passive. A reviewer comments that this overlaps with a different usage in Romance linguistics, which is unfortunate, but I think the current usage is well enough established not to change it.

about the valency meaning constructor, which will contain a dependency on the argument – if we don't, there will be a resource deficit and no successful Glue proof will be possible for the sentence. The template SUPPRESS handles both these tasks:

(39) SUPPRESS(*arg, template*) :=
@NOMAP(*arg*)
$(\uparrow_\sigma$ *arg* REL$) =$ var
@*template*(*arg*)

This template takes two parameters: the argument to be suppressed, and a template name. The first line of SUPPRESS indicates that the argument in question is not mapped to any GF. The second introduces a dummy REL value 'var' for it, in order to satisfy the corresponding existential constraint by providing its s-structure with some content. The template name passed to SUPPRESS as a parameter is applied to the argument being suppressed, and describes how any semantic dependencies are to be resolved.

Perhaps the most straightforward way of resolving the dependency on an argument in the semantics is to existentially close the dependency. This is what the template CLOSURE describes:

(40) CLOSURE(*arg*) :=
$\lambda P.\exists x[P(x)] : [(\uparrow_\sigma$ *arg*$) \multimap \uparrow_\sigma] \multimap \uparrow_\sigma$

The short passive uses this template, for example – the default SUBJ argument is not realised syntactically and is interpreted existentially in the semantics: *The cake was eaten* is truth-conditionally equivalent to *Someone ate the cake*. We can represent the short passive using the SUPPRESS template as follows:

(41) SHORT-PASSIVE :=
@SUPPRESS(%arg1, CLOSURE)

A full template for the English passive will then incorporate both templates:[11]

(42) PASSIVE :=
$(\uparrow$ VOICE$) =$ PASSIVE
$\{$**@**SHORT-PASSIVE | **@**LONG-PASSIVE$\}$

Another way of suppressing an argument in the semantics is to bind its interpretation to another, syntactically realised argument (cf. Alsina 1996: 116ff.). This is what happens in the French reflexive. Grimshaw (1982: 112ff.) gives good reasons

[11] A reviewer complains that having this disjunction misses a generalisation, since "there is just one passive and two ways of realizing the logical subject in the passive". In fact, that is precisely what this formulation shows: there is one PASSIVE template, which contains two additional templates expressing alternative ways of realising the logical subject/more Agent-like argument. There is also a typological significance to dividing the mapping possibilities up in this way: some languages may make use of one but not the other. Although no language has only long passives, there are languages like Latvian which have only short passives (Keenan & Dryer 2007: 331f).

to believe that reflexive sentences like (43) in French are syntactically intransitive, unlike their English translations:

(43) Kira se voit.
Kira REFL *sees*
'Kira sees herself.'

But a verb like *voir* 'see' is semantically a two-place predicate, so in (43) one of its arguments has been suppressed. In this case, its second argument is interpreted as being identical to, or bound by, its first. This is what BIND describes:

(44) $\textsc{Bind}(arg_\beta, arg_\alpha) :=$
$\lambda P \lambda x.P(x)(x) : [(\uparrow_\sigma arg_\alpha) \multimap (\uparrow_\sigma arg_\beta) \multimap \uparrow_\sigma] \multimap (\uparrow_\sigma arg_\beta) \multimap \uparrow_\sigma$

This meaning constructor consumes a dependency on two different arguments and replaces it with a dependency on just one of them, while passing that one argument to the predicate in both of its argument positions. Assuming the clitic *se* contributes a feature [REFL +], we can then capture Grimshaw's (1982) lexical-rule based analysis using the following template instead, which can be added to a transitive verb to turn it into a reflexive:

(45) FR-REFLEXIVE :=
@SUPPRESS(%arg2, BIND(%arg1))
$(\uparrow \textsc{refl}) =_c +$

The first line of the template can be read as 'suppress %arg2 by binding it to %arg1'. Notice that since template parameters are just treated as strings when the template is expanded, we can include complex expressions with some parameters already filled in as the second parameter of SUPPRESS.[12]

7 Conclusion

This paper has presented a new framework for representing claims about the mapping between semantic arguments and grammatical functions, and given a few examples of its application. It differs from previous implementations of LMT in that the formalism which underlies it is vanilla LFG+Glue rather than some additional, novel mechanism. Although I have demonstrated how it can be used to encode at least one version of LMT, the framework is theory-agnostic, and could be applied to different versions of LMT, and in settings which make different architectural assumptions: for example, it is also compatible with a version of LMT which uses a dedicated level of a-structure, provided this takes the form of an AVM, as in Butt et al. (1997). It is my hope that this paper has contributed tools that can be used both to make existing theories more explicit and to enable more transparent comparisons between them, potentially revealing new insights into the data and new perspectives on existing analyses.

[12] I am assuming that **@**TEMPLATE$(X)(Y)$ is intepreted in the same way as **@**TEMPLATE(X, Y).

References

Alsina, Alex. 1996. *The role of argument structure in grammar: evidence from Romance*. CSLI Publications.

Alsina, Alex & Sam A. Mchombo. 1993. Object asymmetries and the Chichewa applicative construction. In Sam A. Mchombo (ed.), *Theoretical aspects of Bantu grammar*, 17–45. CSLI Publications.

Andrews, Avery D. 2008. The role of PRED in LFG+Glue. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG08 Conference*, 46–67. CSLI Publications.

Asudeh, Ash & Gianluca Giorgolo. 2012. Flexible composition for optional and derived arguments. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG12 Conference*, 64–84. CSLI Publications.

Asudeh, Ash, Gianluca Giorgolo & Ida Toivonen. 2014. Meaning and valency. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG14 Conference*, 68–88. CSLI Publications.

Börjars, Kersti, Rachel Nordlinger & Louisa Sadler. 2019. *Lexical-Functional Grammar: an introduction*. Cambridge University Press.

Bresnan, Joan. 1982. The passive in lexical theory. In Joan Bresnan (ed.), *The mental representation of grammatical relations*, 3–86. MIT Press.

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

Bresnan, Joan & Jonni M. Kanerva. 1989. Locative inversion in Chichewa: A case study of factorization in grammar. *Linguistic Inquiry* 20(1). 1–50.

Bresnan, Joan & Annie Zaenen. 1990. Deep unaccusativity in LFG. In Katarzyna Dziwirek, Patrick Farrell & Errapel Mejías Bikandi (eds.), *Grammatical relations: a cross-theoretical perspective*, 45–57. CSLI Publications.

Butt, Miriam. 1995. *The structure of complex predicates in Urdu*. CSLI Publications.

Butt, Miriam, Mary Dalrymple & Anette Frank. 1997. An architecture for linking theory in LFG. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG97 Conference*, CSLI Publications.

Champollion, Lucas. 2015. The interaction of compositional semantics and event semantics. *Linguistics and Philosophy* 38(1). 31–66.

Crouch, Dick, Mary Dalrymple, Ronald M. Kaplan, Tracy Holloway King, John T. Maxwell III & Paula Newman. 2017. *XLE documentation*. Palo Alto Research Center (PARC), Palo Alto, CA.

Dalrymple, Mary, Ronald M. Kaplan & Tracy Holloway King. 2004. Linguistic generalizations over descriptions. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG04 Conference*, 199–208. CSLI Publications.

Dalrymple, Mary, John J. Lowe & Louise Mycock. 2019. *The Oxford reference guide to Lexical Functional Grammar*. Oxford University Press.

Davis, Anthony R. 2011. Thematic roles. In Claudia Maienborn, Klaus von Heusinger & Paul H. Portner (eds.), *Semantics: an international handbook of*

natural language meaning, vol. 1, 399–420. Mouton de Gruyter.
Dowty, David. 1991. Thematic proto-roles and argument selection. *Language* 67(3). 547–619.
Falk, Yehuda N. 2006. *Subjects and Universal Grammar*. Cambridge University Press.
Findlay, Jamie Y. 2016. Mapping theory without argument structure. *Journal of Language Modelling* 4(2). 293–338.
Grimshaw, Jane. 1982. On the lexical representation of Romance reflexive clitics. In Joan Bresnan (ed.), *The mental representation of grammatical relations*, 87–148. MIT Press.
Huddleston, Rodney & Geoffrey K. Pullum. 2002. *The Cambridge grammar of the English language*. Cambridge University Press.
Jackendoff, Ray. 1990. *Semantic structures*. The MIT Press.
Keenan, Edward L. & Matthew S. Dryer. 2007. Passive in the world's languages. In Timothy Shopen (ed.), *Language typology and syntactic description* (2nd edn.). *Volume I: clause structure*, 325–361. Cambridge University Press.
Kibort, Anna. 2001. The Polish passive and impersonal in Lexical Mapping Theory. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG01 Conference*, 163–183. CSLI Publications.
Kibort, Anna. 2007. Extending the applicability of Lexical Mapping Theory. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG07 Conference*, 250–270. CSLI Publications.
Kibort, Anna. 2014. Mapping out a construction inventory with (Lexical) Mapping Theory. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG14 Conference*, 262–282. CSLI Publications.
Kibort, Anna & Joan Maling. 2015. Modelling the syntactic ambiguity of the active vs. passive impersonal in LFG. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG15 Conference*, 145–165. CSLI Publications.
Lovestrand, Joseph. 2018. *Serial verb constructions in Barayin: typology, description and Lexical-Functional Grammar*. D.Phil. thesis, University of Oxford.
Lowe, John J. 2014. Gluing meanings and semantic structures. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG14 Conference*, 387–407. CSLI Publications.
Lowe, John J. 2015. Complex predicates: an LFG + glue analysis. *Journal of Language Modelling* 3(2). 413–462.
Needham, Stephanie & Ida Toivonen. 2011. Derived arguments. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG11 Conference*, 401–421. CSLI Publications.
Rappaport Hovav, Malka & Beth Levin. 2007. Deconstructing thematic hierarchies. In Annie Zaenen, Jane Simpson, Tracy Holloway King, Jane Grimshaw, Joan Maling & Chris Manning (eds.), *Architecture, rules, and preferences: Variations on themes by Joan W. Bresnan*, 385–402. CSLI Publications.