FLEXIBLE COMPOSITION FOR OPTIONAL AND DERIVED ARGUMENTS

Ash Asudeh and Gianluca Giorgolo
Carleton University & University of Oxford
King's College, London & University of Oxford

Proceedings of the LFG12 Conference

Miriam Butt and Tracy Holloway King (Editors)

2012

CSLI Publications

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

Abstract

A simple but insightful analysis of optional and derived arguments at the syntax–semantics interface is provided, based on established features of LFG with Glue Semantics (optionality and templates in lexical entries and flexible, resource-sensitive semantic composition).

1 Introduction[1]

There is broad agreement in linguistic theory that arguments and adjuncts must be distinguished, but there is substantial disagreement as to how the distinction is to be represented and how borderline cases should be captured. There are a number of representational options, of which we list some illustrative examples. In Principles and Parameters Theory (Chomsky 1981, 1995), an argument is either the complement or specifier of a head, whereas an adjunct is adjoined at the XP level. In some versions of Head-Driven Phrase Structure Grammar, an adjunct is distinguished by being a member of the DEPS list but not a member of the VALENCE lists or of the ARG-ST list (Bouma et al. 2001). In LFG, we see a hybrid approach. Adjuncts are distinguished at f-structure by being a member of a predicate's ADJUNCT set, whereas arguments fill specific grammatical functions, such as SUBJ, OBJ, etc. However, given the structure-function mapping principles proposed by Bresnan (2001) and developed further by Toivonen (2001, 2003) (see also Bresnan et al. 2013), adjuncts normally appear in distinguished c-structural positions.

In this paper, we present the initial developments in a theory of adjuncts and arguments, building on recent work by Needham and Toivonen (2011), that uses LFG and Glue Semantics (Dalrymple 1999, 2001, Asudeh 2012) to treat the argument/adjunct distinction not narrowly as an issue of syntactic representation, but rather as a distinction that primarily concerns semantic composition.[2]

The main questions that we seek to answer are the following:

1. What are the implications of optional and derived arguments for the mapping from syntax to semantics?

2. How can lexical generalizations about optional and derived arguments best be captured?

[1]This research was supported by an Early Researcher Award from the Ontario Ministry of Research and Innovation (Asudeh), NSERC Discovery Grant #371969 (Asudeh), SSHRC Standard Research Grant #410-2010-1841 (PI: I. Toivonen; Collaborator: A. Asudeh) and by a grant from the John Fell Oxford University Press Research Fund (Asudeh). We thank the following for helpful comments and questions: Doug Arnold, Boban Arsenijević, Miriam Butt, Mary Dalrymple, Dag Haug, Ron Kaplan, Tracy Holloway King, Stefano Quaglia, Louisa Sadler, Sebastian Sulger, Ida Toivonen, and the participants at LFG12. Any remaining errors are our own.

[2]In the companion piece to this paper (Giorgolo and Asudeh 2012), which also appears in these proceedings, we take a distinct formal approach that uses monads, building on Giorgolo and Asudeh (2011), but we maintain the key insight that the argument/adjunct distinction is an issue of semantic composition.

We attempt initial answers to these questions by looking at three cases:

1. Optional objects of semantically relational verbs (e.g., *drink*, *eat*)
2. Passive *by*-phrases
3. Instrumental *with*-phrases

The paper is organized as follows. Section 2 sets out the phenomena we are interested in and the problems and challenges they constitute. Section 3 presents the key ideas of our analysis informally. Section 4 presents our formal analysis. Section 5 discusses the contribution that templates can make to the analysis. Section 6 concludes.

2 Optional Arguments and Borderline Cases

The problematic cases of interest can be divided into two classes. First, there is the case of predicates that semantically denote a relation (i.e., take two arguments), but which do not require the second argument to be syntactically expressed:

(1) Any child of Kim's is unfortunately likely to drink __.

(2) Kim ate __ at noon.

Clearly one has to drink or eat something, so these verbs are semantically relational, yet the object argument can be unexpressed.

It is typical to contrast verbs like these with similar verbs that do not allow the object to be unexpressed:

(3) a. Isak quaffed his milk at lunch.
b. *Isak quaffed __ at lunch.

(4) a. Thora devoured her cake after dinner.
b. *Thora devoured __ after dinner.

The distinctions between *drink*/*quaff* and *eat*/*devour* need to be captured lexically somehow — in other words, it is part of what we know as language speakers that *drink* can drop its object argument but that *quaff* does not.[3] We refer to these sorts of cases as 'optional arguments'.

Needham and Toivonen (2011) review a number of other cases in which a syntactic phrase seems to be an adjunct in some ways (e.g., it is optional; it is a PP instead of a direct argument), but which seems to be an argument in other ways (e.g., it expresses some entailed participant in the event that the verb denotes). Here are some examples with Needham and Toivonen's labels:

[3] It has been noted (e.g., Jackendoff 2002) that this may be predictable based on semantic factors, since devouring/quaffing is a particular manner of eating/drinking, etc., but this would just seem to mean that the lexical generalization may be stated in a more general fashion, perhaps in a hierarchically organized lexicon, not that it is not part of lexical knowledge.

(5) The hole was plugged by Kim. Passive *by*-phrase

(6) Kim plugged the hole with a cork. Instrumental

(7) Kim's obstruction of the hole Possessive phrase in event nominal

(8) Kim plugged the hole for them. Benefactive

(9) The hole crawled with bugs. Displaced theme

(10) It seemed to Kim like the hole could be plugged. Experiencer

(11) The bugs crawled from the hole. Directional

We follow Needham and Toivonen (2011) in referring to these sorts of cases as 'derived arguments', although it should be fairly obvious that no bright line can be drawn between optional and derived arguments.

2.1 The Problem

The basic intuition behind the argument/adjunct distinction is that arguments are "semantically necessary" in some way that adjuncts are not. As pointed out by Needham and Toivonen (2011), despite the intuitive appeal of this claim, it fails spectacularly because many clear adjuncts, such as those involving time and place, are also clearly semantically necessary: every event that we refer to linguistically happens at some time, in some place. This points to a different understanding of the intuition, which Needham and Toivonen call 'verb specificity': arguments are 'semantically distinctive' in that they are associated with particular verb classes, such that these are distinguished from other classes. Thus, time and place are generally poor arguments, *because* they are ubiquitous and fail to distinguish between verb classes.

The semantic function that arguments play is typically tied to their obligatory realization in syntax, with optionality often taken to be a hallmark of adjuncts. However, there are cases of clear arguments, according to any plausible semantic criterion, which are nevertheless syntactically optional, such as the objects of *drink* and *eat* in English. Similarly, there are argument-like functions ('derived arguments'), such as instrumentals, that distinguish verb classes according to verb specificity, but which seem to always be optional.

Most solutions to this problem can be characterized as some version of the solution of Bresnan (1978), which proposes two distinct versions of, e.g., the verb *eat*.

(12) *eat:* V, [___ NP], NP_1 'eat' NP_2
[___], ($\exists$ y) NP_1 'eat' y

However, this kind of approach is clearly unappealing, because it basically posits an ambiguity for each relevant verb and misses the generalization that, e.g., the 'eating' is the same sort of thing in both cases.

Lastly, it has been noted (e.g., Fillmore 1986) that there may be restrictions on implicit arguments that are absent for their explicit counterparts:

(13) a. Fido ate this morning.
$\Rightarrow$ Whatever Fido ate counts as food for Fido

b. Fido ate my homework.
$\not\Rightarrow$ My homework counts as food for Fido

(14) a. Kim drank last night.
$\Rightarrow$ Whatever Kim drank last night is alcoholic/intoxicating

b. Kim drank milk last night.
$\not\Rightarrow$ Milk is alcoholic/intoxicating

In sum, the challenge is to capture the core argument structure of verb classes that display optional or derived arguments in a way that:

1. Doesn't simply treat distinct valencies as accidentally related (homonymous).
2. Supports a systematic semantic treatment of optional and derived arguments.
3. Enables semantic restrictions on optional arguments to be stated.
4. Captures commonalities between derived arguments and adjuncts

In the next section, we informally sketch our way of meeting this challenge.

3 An Informal Sketch of Our Approach

Our main claim is that a simple but insightful analysis of optional and derived arguments at the syntax–semantics interface can be provided based on established features of Lexical-Functional Grammar with Glue Semantics:

1. **Optionality**, offered by the regular language of LFG's functional descriptions in lexical entries (Kaplan and Bresnan 1982, Dalrymple 2001).
2. **Flexible semantic composition**, offered by the commutative glue logic of Glue Semantics (Dalrymple 1999, 2001, Asudeh 2012).
3. **Resource-sensitive semantic composition**, again offered by the glue logic.
4. **Generalizations over descriptions**, offered by templates (Dalrymple et al. 2004, Asudeh et al. 2008, Asudeh 2012).

The basic strategy will be to break apart lexical information in such a way that, for example, a transitive verb with an optional object can supply semantic information about the implicit object just in case the object is unexpressed. However, a single lexical entry for the verb handles both the intransitive and transitive instantiation of the verb.

We can exemplify the general approach with the following schematized lexical entry for *eat* — as it occurs in the analysis of a sentence like (15) — with most formal details suppressed for now:

(15) Kim ate at noon.

(16) *ate* V ($\uparrow$ PRED) = 'eat'

F-structure constraints

Obligatory Glue meaning constructor;
encodes general semantic information that is
common to transitive and intransitive uses

(**Optional Glue meaning constructor;**
encodes semantic information that is
specific to the intransitive use)

The PRED feature of this lexical entry does not encode whether it is transitive or instransitive. We assume that subcategorization of grammatical functions other than expletives is not represented at f-structure, but is rather captured by resource-sensitive semantic composition (Kuhn 2001, Asudeh 2012). If this were not the case, the formal f-structure description language would force a disjunctive lexical entry — with the attendant issues discussed in section 2.1 — but for theoretically uninteresting reasons (see Giorgolo and Asudeh 2012 for further discussion).

The lexical entry in (16) is different from the disjunctive lexical entries suggested by Bresnan (1978), shown in (12) above, in an important respect. The two Glue meaning constructors in (16) do not stand in a purely disjunctive relationship, whereas the two options in (12) do.[4] In other words, the entry in (16) does not treat the two subcategorizations of *eat* as coincidentally homophonous, but rather posits a single lexical entry with an obligatory meaning constructor that captures the fact that the verb is semantically relational (i.e., it takes two arguments) and posits an optional meaning constructor that existentially closes the second semantic argument if and only if the object is unexpressed. The core meaning of *eat* is thus maintained across the two cases and associated with a single form, in a principled fashion, whereas in (12) it is treated as purely coincidental that the two subcategorizations share the core of their meaning.

4 Analysis

Butt et al. (1997) treat argument structure (a-structure) as a level interpolated between constituent structure and functional structure, such that the correspondence

[4]The logic of the relevant part of the entry in (16) can be represented as $A \veebar (A \wedge B)$, where A is the obligatory meaning constructor, B is the optional meaning constructor, and $\veebar$ is exclusive disjunction. In contrast, the logic of the lexical entry in (12) is purely exclusive disjunction: $A \veebar B$, where A is the transitive option and B is the intransitive option. It is easy to verify that $(A \veebar (A \wedge B)) \not\equiv (A \veebar B)$.

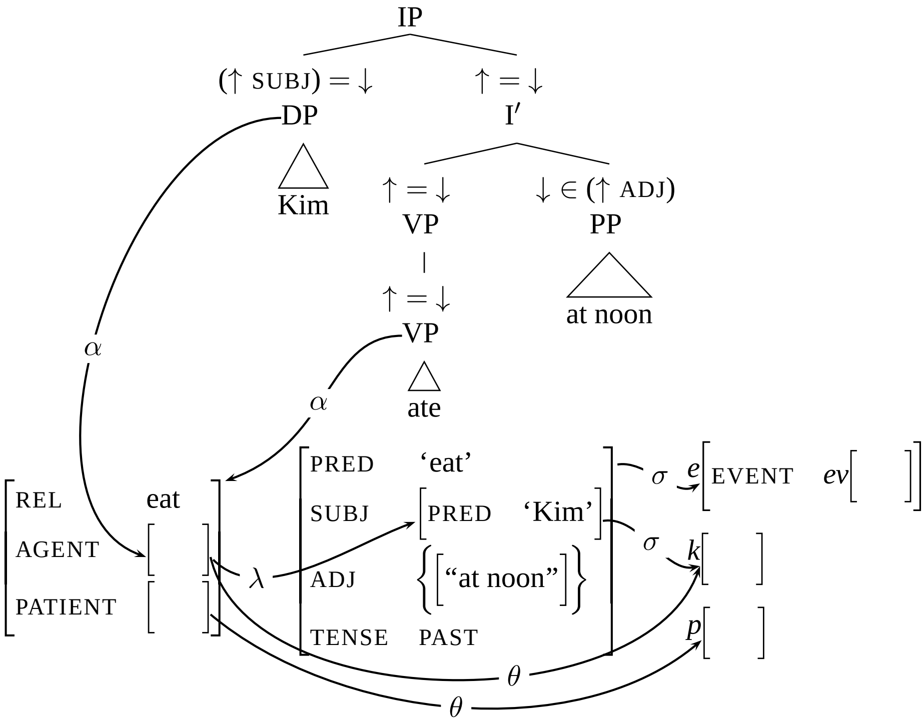

Figure 1: Relevant structures and correspondences for *Kim ate at noon*

function ϕ can be understood as the composition of the correspondence functions α (from c-structure to a-structure) and λ (from a-structure to f-structure). If we adopt this approach, it is necessary to postulate a direct correspondence function θ from argument structure to semantic structure, where the function θ is not the composition of λ and σ.

The example in Figure 1, which adopts the Butt et al. (1997) architecture, illustrates why this is so. The c-structure maps to a-structure, via the α correspondence function. The a-structure maps to f-structure, via the λ correspondence function. Lastly, the f-structure maps to semantic structure, via the σ correspondence function. The PATIENT argument in a-structure must map to an element of semantic structure, since this element provides the resource for the corresponding argument in the semantics in resource-sensitive semantic composition (Dalrymple 2001, Asudeh 2012). However, the PATIENT does not map to an OBJECT grammatical function at f-structure, because this occurrence of *drank* is syntactically intransitive. Therefore, it is not possible to get to the semantic structure correspondent of the PATIENT by going through f-structure, because there is no f-structure correspondent of the PATIENT. Moreover, because the semantic structure is normally unconnected in Glue Semantics (Dalrymple 2001, Asudeh 2012), it is also not possible to get to the semantic structure correspondent of the PATIENT by passing from the outermost a-structure to the outermost f-structure to the semantic structure, since there is no relation expressed in this semantic structure between the semantic structure correspondent of the a-structure containing the PATIENT and the semantic structure correspondent of the PATIENT.

It would be possible to circumvent this problem by positing a null pronominal OBJECT at f-structure, but this is empirically problematic.[5] Standard syntactic tests such as pronominalization, ellipsis, and secondary predication do not support an OBJECT at f-structure when the second argument is unexpressed:

(17) a. Kim drank a beer, but it turned out to be Sandy's.
  b. *Kim drank, but it turned out to be Sandy's.

(18) a. Kim is eating a cake, and so is Sandy. (strict or sloppy)
  b. Kim is eating, and so is Sandy. (sloppy only)

(19) a. Kim drank the whiskey neat.
  b. *Kim drank neat.

Therefore, in order to express the correspondence between the PATIENT and a semantic structure resource, we would have to add a new correspondence function, which we have called θ, to the Butt et al. (1997) architecture.

We instead assume an alternative architecture that does away with the λ-projection, the λ correspondence function, and the θ correspondence function. Argument structure is captured in semantic structure instead. Some of the benefits of this approach are as follows:

1. We achieve a simplified architecture, which eliminates a separate a-structure projection, without losing information.
2. We do not lose linking relations and they are still post-constituent structure.[6]
3. We remove the non-determinacy that results from the presence of both the λ and θ correspondence functions.
4. Many of the meaning constructors for semantic composition are more elegant and simplified.
5. We regain the simple, traditional ϕ mapping from c-structure to f-structure.
6. We gain a connected semantic structure.

Figure 2 shows relevant structures and correspondences for our alternative representation of the example in Figure 1. Since we will be assuming an event semantics for our meaning language, such that thematic roles are functions from events to individuals (Parsons 1990), we avoid redundancy in the argument structure by using

[5]These implicit arguments are therefore not analyzable as cases of "pro-drop", unlike the typologically common case of subject arguments which are not realized in c-structure but which are realized in f-structure.

[6]Feeding argument structure from c-structure is motivated by Butt's (1995) work on Urdu complex predicates, in which she argues that the complex predicates can be syntactically complex in c-structure but nevertheless express a single argument structure like that of a non-complex predicate.

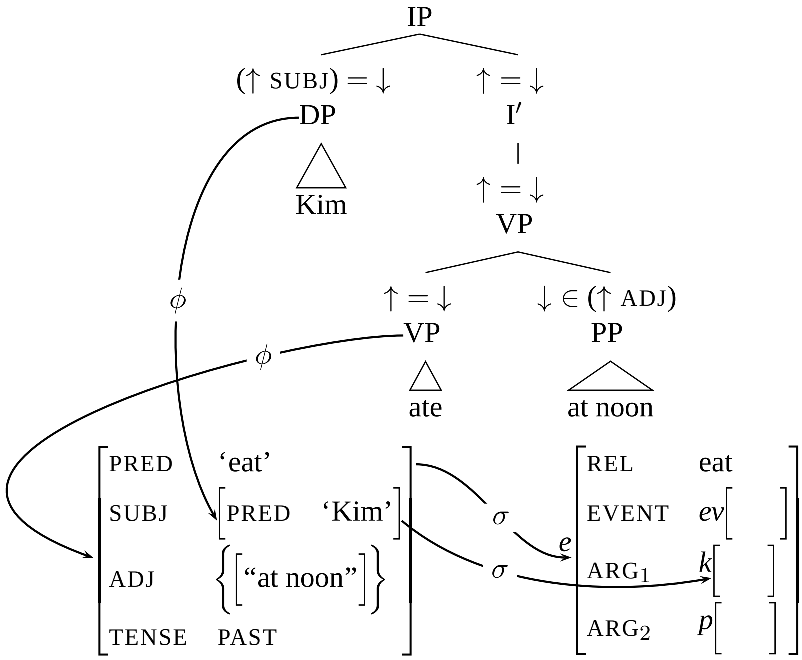

Figure 2: Alternative analysis of *Kim ate at noon.*

attributes like ARG_1 instead of AGENT, etc. If an alternative meaning language that does not encode thematic roles is used, it may be preferable to represent the nature of the arguments directly in a-structure, using the more specific attributes.

4.1 Optional Transitives

Let us begin with the case of ‘optional arguments’, which are semantic arguments that can be syntactically unexpressed, as exemplified by the optional transitivity of *eat* and *drink* versus *devour* and *quaff*. In semantic composition, our analysis simultaneously existentially closes the argument that is alternatively expressed by the object — capturing the fact that even though the argument is unexpressed, it is still an understood argument — and appropriately restricts the existentially closed argument (Fillmore 1986). For example, the existentially closed argument of *drink* is an alcoholic beverage and that of *eat* is food.[7] Moreover, the predicate that expresses this in the semantics must be a relation that also takes the subject as an argument. That is, it is not enough, e.g., for the unexpressed argument to be edible, it must be edible *for* the subject. Contrast the following:

(20) My cousin Kim ate with gusto last night.

(21) My cow Kim ate with gusto last night.

[7]This information is perhaps better treated as a presupposition or conventional implicature than a straight entailment, but we leave this aside here, since it would be straightforward to augment the analysis in standard ways to capture this aspect.

My cousin Kim and my cow Kim eat different sorts of things, and our understanding of these sentences reflects that.[8]

The lexical entry for *ate* is shown in (23) and the Glue proof for example (22) is shown in Figure 6,[9] assuming other standard premises as appropriate and with premises instantiated as per Figure 2 above.

(22) Kim ate at noon.

(23) *ate* V

$(\uparrow \text{PRED}) = \text{`eat'}$
$(\uparrow \text{TENSE}) = \text{PAST}$

$(\uparrow \text{SUBJ})_\sigma = (\uparrow_\sigma \text{ARG}_1)$
$(\uparrow_\sigma \text{ARG}_2)$

$\lambda y \lambda x \lambda e.eat(e) \wedge agent(e) = x \wedge patient(e) = y :$
$(\uparrow_\sigma \text{ARG}_2) \multimap (\uparrow_\sigma \text{ARG}_1) \multimap (\uparrow_\sigma \text{EVENT}) \multimap \uparrow_\sigma$

$$\left(\begin{array}{l} \lambda P \lambda y \exists x.[P(x)(y) \wedge food.for(x,y)] : \\ {[(\uparrow_\sigma \text{ARG}_2) \multimap (\uparrow_\sigma \text{ARG}_1) \multimap \uparrow_\sigma] \multimap [(\uparrow_\sigma \text{ARG}_1) \multimap \uparrow_\sigma]} \end{array} \right)$$

The predicate $food.for(x,y)$ is interpreted such that x is food for y.

The order of arguments of a function can be easily swapped in a Glue proof:

(24)
$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{\lambda y \lambda x.f(x,y) : a \multimap b \multimap c \quad [v : a]^1}{\lambda x.f(x,v) : b \multimap c} \quad [u : b]^2}{f(u,v) : c}}{\lambda v.f(u,v) : a \multimap c} \multimap_{\mathcal{I},1}}{\lambda u \lambda v.f(u,v) : b \multimap a \multimap c} \multimap_{\mathcal{I},2}}{\lambda x \lambda y.f(x,y) : b \multimap a \multimap c} \Rightarrow_\alpha$$

We therefore adopt the convention of choosing a version of the lexically specified function in question that is convenient for the larger proof, abbreviating the function as eat', etc., until the final line of proofs, when the abbreviation is unpacked.

The same lexical entry in (23) is used for the analysis of an example like this:

(25) Kim ate the cake at noon.

However, in this case the resource sensitivity of Glue Semantics (Asudeh 2004, 2012) ensures that the optional premise cannot be selected. The obligatory premise is the only consumer of the object resource in the relevant resource pool. If the

[8] This is not obvious for *drink*, but it seems to be equally the case. For example if Dr. McCoy from *Star Trek* utters "Every subject drank", referring to a group of alien beings in his lab, we expect that each subject drank something compatible with its biology (see also Giorgolo and Asudeh 2012). The editors have mentioned to us that another example is the trolls in Terry Pratchett's *Discworld* novels, who drink stuff which is drinkable only to them.

[9] In order to save space, we gather all Glue proofs at the end of the paper, after the references.

optional premise is also in the resource pool, then the optional premise acts as a modifier of the obligatory premise, as shown in Figure 6 above, such that there is no longer a consumer for the object premise. Therefore, selection of the optional premise leads to a successful Glue proof if and only if there is no object resource. If the object is expressed and therefore contributes a resource, the optional premise is not selected and the obligatory premise consumes its object as per usual. The proof for (25) is shown in Figure 7.

Lastly, let us consider obligatory transitives, such as *devour* and *quaff*, which do not allow their objects to be unexpressed (see (3) and (4) above). The lexical entries for these verbs lack the optional, modificational premise:

(26) *devoured* V
$(\uparrow \text{PRED}) = \text{`devour'}$
$(\uparrow \text{TENSE}) = \text{PAST}$

$(\uparrow \text{SUBJ})_\sigma = (\uparrow_\sigma \text{ARG}_1)$
$(\uparrow \text{OBJ})_\sigma = (\uparrow_\sigma \text{ARG}_2)$

$\lambda y \lambda x \lambda e.devour(e) \ \wedge \ agent(e) = x \ \wedge \ patient(e) = y :$
$(\uparrow_\sigma \text{ARG}_2) \multimap (\uparrow_\sigma \text{ARG}_1) \multimap (\uparrow_\sigma \text{EVENT}) \multimap \uparrow_\sigma$

Resource-sensitive composition ensures that predicates like this must have an expressed object that contributes the ARG_2 resource; otherwise the dependency on this resource is not properly discharged and there is no valid Glue proof.

4.1.1 Scope

Fodor and Fodor (1980) note that a quantifier in subject position must take wide scope over the existentially closed implicit argument of a syntactically intransitive but semantically relational verb:[10]

(27) Every student ate.
$\Rightarrow$ For every student x, there is some thing y such that x ate y.
$\not\Rightarrow$ There is some thing y such that, for every student x, x ate y.

Our analysis captures this scope generalization. The quantifier and the optional premise contributed by the verb *ate* both constitute dependencies on a dependency on the subject. That is, both the quantifier and the optional premise are consumers of a premise that can be schematized as $subj \multimap predicate$. There is only one such premise (the verb's premise, having consumed the implicit argument's resource). The optional premise, however, is a modifier-type premise that outputs the same dependency again. Therefore, the quantifier can consume the output of the optional premise. In contrast, the quantifier does not output a premise of this type, but rather one of a propositional type. Therefore, the optional premise cannot consume the

[10]This claim has been refined by Lasersohn (1993), based on distributed readings, but he does not seem to have found the correct generalization. This is discussed further in Giorgolo and Asudeh (2012).

output of the quantifier. This means that the quantifier must come later in the proof, which entails that it scopes wide. The successful proof for the wide scope reading is shown in Figure 8.[11]

4.2 Passives

We assume that, in the absence of a *by*-phrase, the suppressed argument of a passive is not represented at f-structure, but is represented at semantic structure. A short passive is thus semantically relational, but syntactically intransitive, much like our previous cases. We again propose a lexical entry that has an obligatory semantic component and an optional semantic component. The suppressed argument is again optionally existentially closed. Given Glue's resource-sensitive semantic composition, this option only leads to a well-formed proof in the absence of a *by*-phrase. If both the meaning constructor contributed by a *by*-phrase and the optional existential meaning constructor were present, there would be two dependencies on the resource corresponding to the suppressed highest role of the predicate (e.g., the $\textsc{arg}_1$ of *eaten*, which is mapped to the *by*-phrase, if one is present), but only one such dependency could be satisfied. Resource sensitivity similarly guarantees that the optional premise contributed by the passive verb *must* be realized in the absence of a *by*-phrase, because otherwise the dependency on the highest role (i.e., $\textsc{arg}_1$) is not discharged. We thus correctly predict that there is existential closure of the suppressed argument if and only if there is no *by*-phrase.

Consider the examples of a short passive and a *by*-passive in (28) and (29).

(28) Kim was eaten last night.

(29) Kim was eaten by Godzilla last night.

Lexical entries for the passive predicate, *eaten*, and the passive *by* are shown in (30) and (31).[12]

(30) *eaten* V

$(\uparrow \textsc{pred}) = \text{`eat'}$
$(\uparrow \textsc{voice}) = \textsc{passive}$

$(\uparrow \textsc{subj})_\sigma = (\uparrow_\sigma \textsc{arg}_2)$
$(\uparrow_\sigma \textsc{arg}_1)$

$\lambda x \lambda y \lambda e.eat(e) \wedge agent(e) = x \wedge patient(e) = y :$
$(\uparrow_\sigma \textsc{arg}_1) \multimap (\uparrow_\sigma \textsc{arg}_2) \multimap (\uparrow_\sigma \textsc{event}) \multimap \uparrow_\sigma$

$\big(\lambda P \exists x.[P(x)] : [(\uparrow_\sigma \textsc{arg}_1) \multimap \uparrow_\sigma] \multimap \uparrow_\sigma \big)$

[11]Our approach allows the subject quantifier and existential event closure to scope freely with respect to each other, since examples like (27) are ambiguous between a single event of every student eating and separate events of each student eating. The proof in Figure 8 captures only the first of these readings.

[12] There has been some inconsistency in the LFG literature regarding the realization of the *by*-phrase at f-structure: Is it an ADJ or an OBL? (See Needham and Toivonen 2011 for discussion and references.) This choice does not substantively affect our analysis, but we assume the *by*-phrase is an OBL here. Otherwise, change the occurrences of $(\uparrow \textsc{obl})$ in (31) to $(\textsc{adj} \in \uparrow)$.

(31) *by* P $(\uparrow \textsc{pred}) = \text{`by'}$
$((\textsc{obl}\uparrow)\ \textsc{voice}) =_c \textsc{passive}$
$(\uparrow \textsc{obj})_\sigma = ((\textsc{obl}\uparrow)_\sigma\ \textsc{arg}_1)$
$\lambda x \lambda P.[P(x)] : (\uparrow \textsc{obj})_\sigma \multimap [\uparrow_\sigma\ \multimap\ (\textsc{obl}\uparrow)_\sigma] \multimap (\textsc{obl}\uparrow)_\sigma$

Rather than existentially closing the suppressed argument of its passive verb, the *by*-phrase saturates the corresponding argument of the passive with the OBJ in the *by*-phrase (e.g., *Godzilla* in *by Godzilla*). Needham and Toivonen (2011) also note that the nominal in the *by*-phrase must fill the role of whatever was the highest/suppressed argument of the verb that it modifies; this is accomplished through the equation $(\uparrow \textsc{obj})_\sigma = ((\textsc{obl}\uparrow)_\sigma\ \textsc{arg}_1)$ in the lexical entry in (31).

Figures 3 and 4 respectively show analyses of examples (28) and (29). From this point on, for reasons of space, we do not show c-structures, since they can be inferred from f-structures. Figures 9 and 10 show Glue proofs for the examples.

4.3 Instrumentals

The last case we consider is instrumental *with*-phrases:

(32) a. Robin killed Sandy.
b. Robin killed Sandy with dynamite.

(33) a. An explosion killed Sandy.
b. #An explosion killed Sandy with dynamite.

Instrumental *with*-phrases, like passive *by*-phrases, are instances of Needham and Toivonen's ‘derived arguments’.

Following Reinhart (2002), Needham and Toivonen (2011: 415) note that instrumental *with*-phrases are only well-formed with “agent verbs”, cf. (32b) vs. (33b). Stated as such, a generalization seems to be missed, because we seem to require two verbs *kill*: agentive *kill*$_1$ and non-agentive *kill*$_2$. Our analysis avoids this undesirable outcome while properly capturing the empirical generalization.

We capture the contrast through the same kind of standard restrictive semantics used for *eat* above, by imposing a requirement of animacy on the subject argument while simultaneously adding the information that the object of the *with*-phrase is

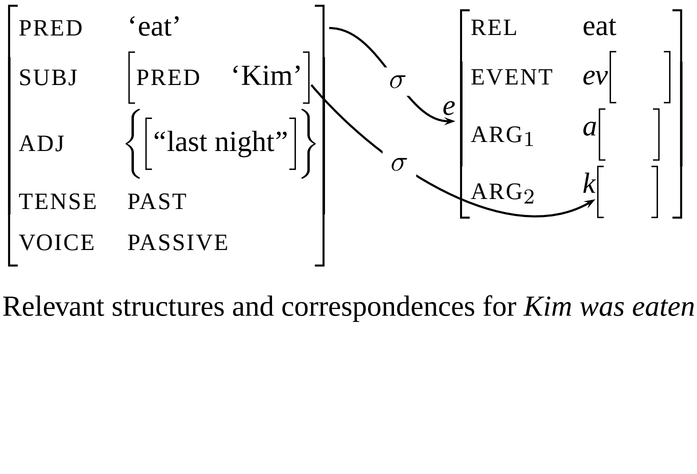

Figure 3: Relevant structures and correspondences for *Kim was eaten last night.*

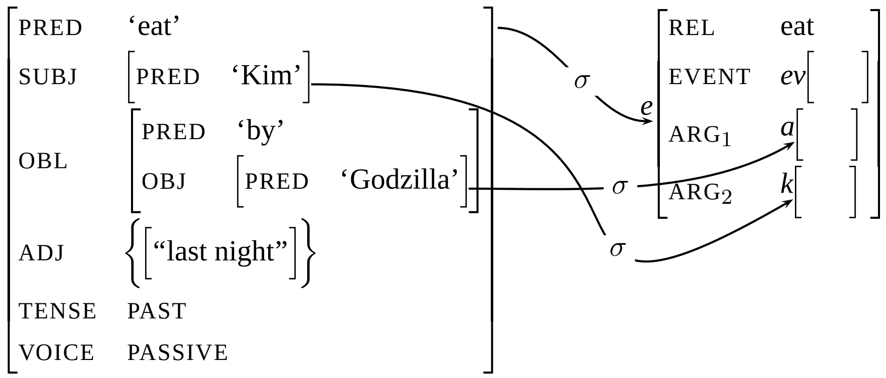

Figure 4: Relevant structures and correspondences for *Kim was eaten by Godzilla last night.*

an instrument in the event. We assume that, unlike passive *by*-phrases, instrumental *with*-phrases add an argument that is not otherwise linguistically represented for the predicate, thus constituting central cases of 'derived arguments'. However, this is accomplished through lexical information associated with instrumental *with*, rather than by directly modifying the lexical entry of the verb.[13] Consider example (34) in light of the lexical entry in (35).

(34) Kim tapped Sandy with Excalibur.

(35) *with* P

$(\uparrow \text{PRED}) = \text{`with'}$

$(\uparrow \text{OBJ})_\sigma = ((\text{OBL} \uparrow)_\sigma \text{ INSTRUMENT})$

$$\lambda y \lambda P \lambda x \lambda e.[P(x)(e) \;\wedge\; \mathit{animate}(x) \;\wedge\; \mathit{instrument}(e) = y] : \\ (\uparrow \text{OBJ})_\sigma \multimap \\ [((\text{OBL} \uparrow) \text{ SUBJ})_\sigma \multimap ((\text{OBL} \uparrow)_\sigma \text{ EVENT}) \multimap (\text{OBL} \uparrow)_\sigma] \multimap \\ ((\text{OBL} \uparrow) \text{ SUBJ})_\sigma \multimap ((\text{OBL} \uparrow)_\sigma \text{ EVENT}) \multimap (\text{OBL} \uparrow)_\sigma$$

No mention is made of the thematic role of the subject, allowing it to be the same role whether the instrumental is present or not. The f-structure and semantic structure for (34) are shown in Figure 5 and the Glue proof is shown in Figure 11.

5 Further Capturing Lexical Generalizations

An LFG template or macro is an abbreviation for a set of equations or constraints (Dalrymple et al. 2004, Asudeh et al. 2008, Crouch et al. 2011, Asudeh 2012). A template is referenced in a lexical entry, as denoted by the '@' prefix. The semantics of template invocation is simple substitution: any template in a lexical

[13]Once again, for consistency with Needham and Toivonen (2011), we treat the *with*-phrase as an OBL, but once again this does not substantively affect our analysis; see footnote 12.

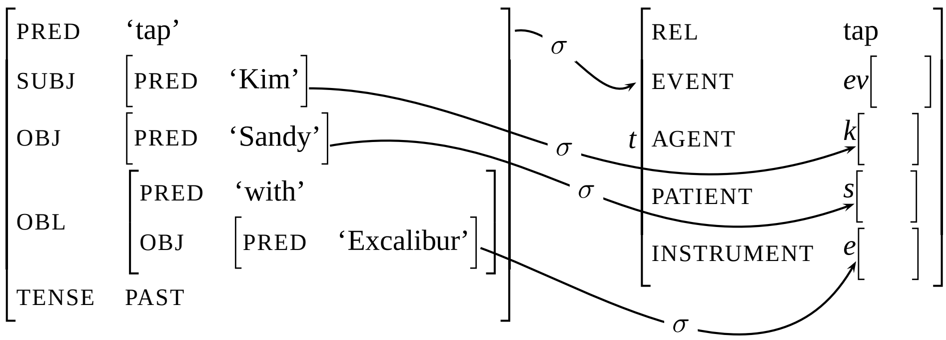

Figure 5: Relevant structures and correspondences for *Kim tapped Sandy with Excalibur.*

entry can be equivalently replaced by the contents of template. Even though they are purely abbreviatory devices, templates can capture linguistic generalizations, since they cross-classify the lexical entries that contain the same templates. Thus, even though a grammar with templates is extensionally equivalent to a grammar with all template calls substituted with the contents of the templates, the former grammar might express generalizations that the latter does not.

The cases that we have examined demonstrate this. It is clear that there is something common to semantically relational verbs — e.g., *eat*, *drink*, *devour*, and *quaff* — and it is also clear that these verbs further subcategorize into the optionally transitive — e.g., *eat* and *drink* — versus the obligatorily transitive — e.g., *devour*, and *quaff*. The following templates and lexical entries demonstrate how templates can capture such generalizations:

(36) $\textsc{past} = \ (\uparrow \textsc{tense}) = \textsc{past}$

(37) $\textsc{agent-patient-verb} =$

$(\uparrow \textsc{subj})_\sigma = (\uparrow_\sigma \textsc{arg}_1)$

$\lambda P \lambda y \lambda x \lambda e. P(e) \ \wedge \ agent(e) = x \ \wedge \ patient(e) = y :$
$[(\uparrow_\sigma \textsc{event}) \multimap \uparrow_\sigma] \multimap (\uparrow_\sigma \textsc{arg}_2) \multimap (\uparrow_\sigma \textsc{arg}_1) \multimap (\uparrow_\sigma \textsc{event}) \multimap \uparrow_\sigma$

(38) $\textsc{optional-transitive} = \ (\uparrow_\sigma \textsc{arg}_2)$

$\lambda P \exists x.[P(x)] : [(\uparrow_\sigma \textsc{arg}_2) \multimap \uparrow_\sigma] \multimap \uparrow_\sigma$

(39) *ate* V $(\uparrow \textsc{pred}) = \text{'eat'}$
@PAST
@AGENT-PATIENT-VERB

$$\left(\begin{array}{l} @\textsc{optional-transitive} \\ \lambda P \lambda y \lambda x \lambda e. P(y)(x)(e) \wedge food.for(y,x) : \\ {[(\uparrow_\sigma \textsc{arg}_2) \multimap (\uparrow_\sigma \textsc{arg}_1) \multimap (\uparrow_\sigma \textsc{event}) \multimap \uparrow_\sigma] \multimap} \\ \quad (\uparrow_\sigma \textsc{arg}_2) \multimap (\uparrow_\sigma \textsc{arg}_1) \multimap (\uparrow_\sigma \textsc{event}) \multimap \uparrow_\sigma \end{array} \right)$$

$\lambda e. eat(e) : (\uparrow_\sigma \textsc{event}) \multimap \uparrow_\sigma$

(40) *devoured* V $(\uparrow \text{PRED}) = \text{‘devour’}$
@PAST
@AGENT-PATIENT-VERB
$(\uparrow \text{OBJ})_\sigma = (\uparrow_\sigma \text{ARG}_2)$
$\lambda e.devour(e) : (\uparrow_\sigma \text{EVENT}) \multimap \uparrow_\sigma$

(41) PASSIVE = $(\uparrow \text{VOICE}) = \text{PASSIVE}$
$(\uparrow \text{SUBJ})_\sigma = (\uparrow_\sigma \text{ARG}_2)$
$(\uparrow_\sigma \text{ARG}_1)$
$(\ \lambda P \exists x.[P(x)] : [(\uparrow_\sigma \text{ARG}_1) \multimap \uparrow_\sigma] \multimap \uparrow_\sigma\ )$

(42) *eaten* V $(\uparrow \text{PRED}) = \text{‘eat’}$
@PASSIVE
$\lambda x \lambda y \lambda e.eat(e) \ \wedge\ agent(e) = x \ \wedge\ patient(e) = y :$
$(\uparrow_\sigma \text{ARG}_1) \multimap (\uparrow \text{SUBJ})_\sigma \multimap (\uparrow_\sigma \text{EVENT}) \multimap \uparrow_\sigma$

Reasons of space preclude us from discussing these entries carefully, but it should be evident that much information has been moved out of particular lexical entries into templates that generalize across lexical entries. We do note that the Glue meaning constructors have been modified as a result of the new distribution of information, which further highlights the flexibility of resource-sensitive composition in Glue Semantics. One welcome result of this modification is that the core, obligatory information of verbs is now just a predicate on events, such that each verb adds very little parochial information. This can also be readily extended for verbs like *devour* such that the core meaning involves an eating event and an appropriate intensifying adverbial that is a manner modifier of the eating event.

6 Conclusion

We sought to answer two main questions about optional and derived arguments, and to meet a number of challenges constituted by these phenomena. The first question concerned implications for the syntax–semantics interface. We have presented an analysis which merges argument structure and semantic structure and which depends on flexible composition in Glue Semantics. This flexibility ultimately derives from the fact that Glue is a type-logical approach that separates syntax and semantics, very much in the spirit of LFG, such that the logic of composition is commutative. The second question concerned how to properly capture lexical generalizations about the relevant cases. Flexible composition again featured here: lexical entries can contribute obligatory, core meaning constructors as well as optional, modificational meaning constructors, where the optionality is captured by LFG's normal language of lexical specification. LFG templates can capture yet further lexical generalization.

Our analysis meets various general challenges to analyses of these phenomena. First, we do not treat the distinct valencies of the predicates in question as ambiguities (accidental homonymy), but rather as involving core information and modificational information, which interacts properly with optionality. This modificational information is intuitively and formally adjunct-like, which perhaps sheds some light on why these cases have adjunct-like behaviour. It also enables semantic restrictions on optional arguments to be captured. Lastly, resource-sensitive semantic composition in Glue Semantics ensures that the obligatory and optional information interact properly.

In conclusion, this sort of approach can constitute the first step toward a more general theory of arguments and adjuncts, although we are not necessarily wed to all of the formal details. In the companion paper in this volume Giorgolo and Asudeh (2012), we present a different way to deal with the issue of unexpressed arguments in a resource-sensitive semantics.

References

Asudeh, Ash. 2004. Resumption as Resource Management. Ph.D. thesis, Stanford University.

—. 2012. *The Logic of Pronominal Resumption*. Oxford: Oxford University Press.

Asudeh, Ash, Mary Dalrymple, and Ida Toivonen. 2008. Constructions with Lexical Integrity: Templates as the Lexicon-Syntax Interface. In Miriam Butt and Tracy Holloway King, eds., *Proceedings of the LFG08 Conference*, 68–88. Stanford, CA: CSLI Publications.

Bouma, Gosse, Robert Malouf, and Ivan A. Sag. 2001. Satisfying Constraints on Extraction and Adjunction. *Natural Language and Linguistic Theory* 19: 1–65.

Bresnan, Joan. 1978. A Realistic Transformational Grammar. In Morris Halle, Joan Bresnan, and George A. Miller, eds., *Linguistic Theory and Psychological Reality*, 1–59. Cambridge, MA: MIT Press.

—. 2001. *Lexical-Functional Syntax*. Oxford: Blackwell.

Bresnan, Joan, Ash Asudeh, Ida Toivonen, and Stephen Wechsler. 2013. *Lexical-Functional Syntax*. Oxford: Wiley-Blackwell, 2nd edn. Forthcoming.

Butt, Miriam. 1995. *The Structure of Complex Predicates in Urdu*. Stanford, CA: CSLI Publications.

Butt, Miriam, Mary Dalrymple, and Anette Frank. 1997. An Architecture for Linking Theory in LFG. In Miriam Butt and Tracy Holloway King, eds., *Proceedings of the LFG97 Conference*. Stanford, CA: CSLI Publications.

Butt, Miriam, and Tracy Holloway King, eds. 2011. *Proceedings of the LFG11 Conference*. Stanford, CA: CSLI Publications.

Chomsky, Noam. 1981. *Lectures on Government and Binding*. Dordrecht: Foris.

—. 1995. *The Minimalist Program*. Cambridge, MA: MIT Press.

Crouch, Richard, Mary Dalrymple, Ronald M. Kaplan, Tracy King, John T. Maxwell, III, and Paula Newman. 2011. *XLE Documentation*. Palo Alto Research Center (PARC), Palo Alto, CA.

Dalrymple, Mary, ed. 1999. *Semantics and Syntax in Lexical Functional Grammar: The Resource Logic Approach*. Cambridge, MA: MIT Press.

Dalrymple, Mary. 2001. *Lexical Functional Grammar*. San Diego, CA: Academic Press.

Dalrymple, Mary, Ronald M. Kaplan, and Tracy Holloway King. 2004. Linguistic Generalizations over Descriptions. In Miriam Butt and Tracy Holloway King, eds., *Proceedings of the LFG04 Conference*, 199–208. Stanford, CA: CSLI Publications.

Dalrymple, Mary, Ronald M. Kaplan, John T. Maxwell III, and Annie Zaenen, eds. 1995. *Formal Issues in Lexical-Functional Grammar*. Stanford, CA: CSLI Publications.

Fillmore, Charles J. 1986. Pragmatically Controlled Zero Anaphora. In *Proceedings of the Twelfth Annual Meeting of the Berkeley Linguistics Society*, 95–107. Berkeley: Berkeley Linguistics Society.

Fodor, Jerry A., and Janet Dean Fodor. 1980. Functional Structure, Quantifiers, and Meaning Postulates. *Linguistic Inquiry* 11(4): 759–770.

Giorgolo, Gianluca, and Ash Asudeh. 2011. Multidimensional Semantics with Unidimensional Glue Logic. In Butt and King 2011, 236–256.

—. 2012. Missing Resources in a Resource-Sensitive Semantics. In Miriam Butt and Tracy Holloway King, eds., *Proceedings of the LFG12 Conference*, This volume. Stanford, CA: CSLI Publications.

Jackendoff, Ray. 2002. *Foundations of Language: Brain, Meaning, Grammar, Evolution*. Oxford: Oxford University Press.

Kaplan, Ronald M., and Joan Bresnan. 1982. Lexical-Functional Grammar: A Formal System for Grammatical Representation. In Joan Bresnan, ed., *The Mental Representation of Grammatical Relations*, 173–281. Cambridge, MA: MIT Press. Reprinted in Dalrymple et al. (1995: 29–135).

Kuhn, Jonas. 2001. Resource Sensitivity in the Syntax-Semantics Interface and the German Split NP Construction. In W. Detmar Meurers and Tibor Kiss, eds., *Constraint-Based Approaches to Germanic Syntax*. Stanford, CA: CSLI Publications.

Lasersohn, Peter N. 1993. Lexical Distributivity and Implicit Arguments. In *Proceedings from Semantics and Linguistic Theory III*, 145–161.

Needham, Stephanie, and Ida Toivonen. 2011. Derived Arguments. In Butt and King 2011, 401–421.

Parsons, Terence. 1990. *Events in the Semantics of English: A Study in Subatomic Semantics*. Cambridge, MA: MIT Press.

Reinhart, Tanya. 2002. The Theta System — An Overview. *Theoretical Linguistics* 28: 229–290.

Toivonen, Ida. 2001. The Phrase Structure of Non-Projecting Words. Ph.D. thesis, Stanford University.

—. 2003. *Non-Projecting Words: A Case Study of Swedish Verbal Particles*. Dordrecht: Kluwer.

ate (opt.)
$$\lambda P \lambda y \exists x.[P(x)(y) \ \wedge \ food.for(x,y)] : [p \multimap k \multimap e] \multimap k \multimap e$$

ate
$$\dfrac{eat' : ev \multimap p \multimap k \multimap e \qquad [e' : ev]^1}{eat'(e') : p \multimap k \multimap e}$$

$$\dfrac{\lambda P \lambda y \exists x.[P(x)(y) \wedge food.for(x,y)] : [p \multimap k \multimap e] \multimap k \multimap e \qquad eat'(e') : p \multimap k \multimap e}{\lambda y \exists x.[eat'(e')(x)(y) \wedge food.for(x,y)] : k \multimap e}$$

Kim
$$kim : k$$

$$\dfrac{\lambda y \exists x.[eat'(e')(x)(y) \wedge food.for(x,y)] : k \multimap e \qquad kim : k}{\exists x.[eat'(e')(x)(kim) \wedge food.for(x,kim)] : e}$$

$$\dfrac{\exists x.[eat'(e')(x)(kim) \wedge food.for(x,kim)] : e}{\lambda e' \exists x.[eat'(e')(x)(kim) \wedge food.for(x,kim)] : ev \multimap e} \multimap_{\mathcal{I},1}$$

at noon
$$\lambda P \lambda e''.[P(e'') \wedge at.noon(e'')] : (ev \multimap e) \multimap (ev \multimap e)$$

$$\dfrac{\lambda P \lambda e''.[P(e'') \wedge at.noon(e'')] : (ev \multimap e) \multimap (ev \multimap e) \qquad \lambda e' \exists x.[eat'(e')(x)(kim) \wedge food.for(x,kim)] : ev \multimap e}{\lambda e'' \exists x.[eat'(e'')(x)(kim) \wedge food.for(x,kim) \wedge at.noon(e'')] : ev \multimap e}$$

PAST
$$\lambda P \exists e.[P(e) \wedge past(e)] : (ev \multimap e) \multimap e$$

$$\dfrac{\lambda P \exists e.[P(e) \wedge past(e)] : (ev \multimap e) \multimap e \qquad \lambda e'' \exists x.[eat'(e'')(x)(kim) \wedge food.for(x,kim) \wedge at.noon(e'')] : ev \multimap e}{\exists e \exists x.[eat'(e)(x)(kim) \wedge food.for(x,kim) \wedge at.noon(e) \wedge past(e)] : e}$$

$$\dfrac{\exists e \exists x.[eat'(e)(x)(kim) \wedge food.for(x,kim) \wedge at.noon(e) \wedge past(e)] : e}{\exists e \exists x.[eat(e) \wedge agent(e) = kim \wedge patient(e) = x \wedge food.for(x,kim) \wedge at.noon(e) \wedge past(e)] : e} \Rightarrow_\beta$$

Figure 6: Proof for *Kim ate at noon.*

ate
$$eat' : c \multimap k \multimap ev \multimap e$$

the cake
$$\iota x.[cake(x)] : c$$

$$\dfrac{eat' : c \multimap k \multimap ev \multimap e \qquad \iota x.[cake(x)] : c}{eat'(\iota x.[cake(x)]) : k \multimap ev \multimap e}$$

Kim
$$kim : k$$

$$\dfrac{eat'(\iota x.[cake(x)]) : k \multimap ev \multimap e \qquad kim : k}{eat'(\iota x.[cake(x)])(kim) : ev \multimap e}$$

at noon
$$\lambda P \lambda e'.[P(e') \wedge at.noon(e')] : (ev \multimap e) \multimap (ev \multimap e)$$

$$\dfrac{\lambda P \lambda e'.[P(e') \wedge at.noon(e')] : (ev \multimap e) \multimap (ev \multimap e) \qquad eat'(\iota x.[cake(x)])(kim) : ev \multimap e}{\lambda e'.[eat'(\iota x.[cake(x)])(kim)(e') \wedge at.noon(e')] : ev \multimap e}$$

PAST
$$\lambda P \exists e.[P(e) \wedge past(e)] : (ev \multimap e) \multimap e$$

$$\dfrac{\lambda P \exists e.[P(e) \wedge past(e)] : (ev \multimap e) \multimap e \qquad \lambda e'.[eat'(\iota x.[cake(x)])(kim)(e') \wedge at.noon(e')] : ev \multimap e}{\exists e.[eat'(\iota x.[cake(x)])(kim)(e) \wedge at.noon(e) \wedge past(e)] : e}$$

$$\dfrac{\exists e.[eat'(\iota x.[cake(x)])(kim)(e) \wedge at.noon(e) \wedge past(e)] : e}{\exists e.[eat(e) \wedge agent(e) = kim \wedge patient(e) = \iota x.[cake(x)] \wedge at.noon(e) \wedge past(e)] : e} \Rightarrow_\beta$$

Figure 7: Proof for *Kim ate the cake at noon.*

$$\dfrac{\textbf{ate}\;\; eat' : ev \multimap p \multimap s \multimap e \qquad [e' : ev]^1}{eat'(e') : p \multimap s \multimap e}$$

$$\dfrac{\textbf{ate (opt.)}\;\; \lambda P \lambda y \exists x.[P(x)(y) \;\wedge\; food.for(x,y)] : [p \multimap s \multimap e] \multimap s \multimap e \qquad eat'(e') : p \multimap s \multimap e}{\lambda y \exists x.[eat'(e')(x)(y) \wedge food.for(x,y)] : s \multimap e}$$

$$\dfrac{\textbf{every student}\;\; \lambda P \forall z.[student(z) \rightarrow \; P(z)] : \forall X.[(s \multimap X) \multimap X] \qquad \lambda y \exists x.[eat'(e')(x)(y) \wedge food.for(x,y)] : s \multimap e}{\forall z.[student(z) \rightarrow \; \exists x.[eat'(e')(x)(z) \wedge food.for(x,z)]] : e} \; \forall_{\mathcal{E}}[e/X]$$

$$\dfrac{\forall z.[student(z) \rightarrow \; \exists x.[eat'(e')(x)(z) \wedge food.for(x,z)]] : e}{\lambda e' \forall z.[student(z) \rightarrow \; \exists x.[eat'(e')(x)(z) \wedge food.for(x,z)]] : ev \multimap e} \multimap_{\mathcal{I},1}$$

$$\dfrac{\textbf{PAST}\;\; \lambda P \exists e.[P(e) \wedge past(e)] : (ev \multimap e) \multimap e \qquad \lambda e' \forall z.[student(z) \rightarrow \; \exists x.[eat'(e')(x)(z) \wedge food.for(x,z)]] : ev \multimap e}{\exists e.[\forall z.[student(z) \rightarrow \; \exists x.[eat'(e'')(x)(z) \wedge food.for(x,z)]] \wedge at.noon(e'') \wedge past(e)] : e}$$

$$\dfrac{\exists e.[\forall z.[student(z) \rightarrow \; \exists x.[eat'(e'')(x)(z) \wedge food.for(x,z)]] \wedge at.noon(e'') \wedge past(e)] : e}{\exists e.[\forall z.[student(z) \rightarrow \; \exists x.[eat(e) \wedge agent(e) = z \wedge patient(e) = x \wedge food.for(x,z)]] \wedge at.noon(e) \wedge past(e)] : e} \Rightarrow_{\beta}$$

Figure 8: Proof for subject wide scope reading of *Every student ate*.

$$\dfrac{\textbf{eaten}\;\; eat' : ev \multimap k \multimap a \multimap e \qquad [e' : ev]^1}{eat'(e) : k \multimap a \multimap e}$$

$$\dfrac{eat'(e) : k \multimap a \multimap e \qquad \textbf{Kim}\;\; kim : k}{eat'(e')(kim) : a \multimap e}$$

$$\dfrac{\textbf{eaten (opt.)}\;\; \lambda P \exists x.[P(x)] : (a \multimap e) \multimap e \qquad eat'(e')(kim) : a \multimap e}{\exists x.[eat'(e')(kim)(x)] : e}$$

$$\dfrac{\exists x.[eat'(e')(kim)(x)] : e}{\lambda e' \exists x.[eat'(e')(kim)(x)] : ev \multimap e} \multimap_{\mathcal{I},1}$$

$$\dfrac{\textbf{last night}\;\; \lambda P \lambda e''.[P(e'') \wedge last.night(e'')] : (ev \multimap e) \multimap (ev \multimap e) \qquad \lambda e' \exists x.[eat'(e')(kim)(x)] : ev \multimap e}{\lambda e'' \exists x.[eat'(e')(kim)(x) \wedge last.night(e'')] : ev \multimap e}$$

$$\dfrac{\textbf{was}\;\; \lambda P \exists e.[P(e) \wedge past(e)] : (ev \multimap e) \multimap e \qquad \lambda e'' \exists x.[eat'(e')(kim)(x) \wedge last.night(e'')] : ev \multimap e}{\exists e \exists x.[eat'(e)(kim)(x) \wedge last.night(e) \wedge past(e)] : e}$$

$$\dfrac{\exists e \exists x.[eat'(e)(kim)(x) \wedge last.night(e) \wedge past(e)] : e}{\exists e \exists x.[eat(e) \wedge agent(e) = x \wedge patient(e) = kim \wedge last.night(e) \wedge past(e)] : e} \Rightarrow_{\beta}$$

Figure 9: Proof for *Kim was eaten last night.*

$$\dfrac{\dfrac{\textbf{by}\ \ \lambda x\lambda P.[P(x)] : g \multimap (a \multimap e) \multimap e \qquad \textbf{Godzilla}\ \ godzilla : g}{\lambda P.[P(godzilla)] : (a \multimap e) \multimap e} \qquad \dfrac{\dfrac{\textbf{eaten}\ \ eat' : ev \multimap k \multimap a \multimap e \qquad [e' : ev]^1}{eat'(e) : k \multimap a \multimap e} \qquad \textbf{Kim}\ \ kim : k}{eat'(e')(kim) : a \multimap e}}{eat'(e')(kim)(godzilla) : e}$$

$$\dfrac{eat'(e')(kim)(godzilla) : e}{\lambda e'.[eat'(e')(kim)(godzilla)] : ev \multimap e}\multimap_{\mathcal{I},1}$$

$$\dfrac{\textbf{last night}\ \ \lambda P\lambda e''.[P(e'') \wedge last.night(e'')] : (ev \multimap e) \multimap (ev \multimap e) \qquad \lambda e'.[eat'(e')(kim)(godzilla)] : ev \multimap e}{\lambda e''.[eat'(e')(kim)(godzilla) \wedge last.night(e'')] : ev \multimap e}$$

$$\dfrac{\textbf{was}\ \ \lambda P\exists e.[P(e) \wedge past(e)] : (ev \multimap e) \multimap e \qquad \lambda e''.[eat'(e')(kim)(godzilla) \wedge last.night(e'')] : ev \multimap e}{\exists e.[eat'(e)(kim)(godzilla) \wedge last.night(e) \wedge past(e)] : e}$$

$$\dfrac{\exists e.[eat'(e)(kim)(godzilla) \wedge last.night(e) \wedge past(e)] : e}{\exists e.[eat(e) \wedge agent(e) = godzilla \wedge patient(e) = kim \wedge last.night(e) \wedge past(e)] : e}\Rightarrow_\beta$$

Figure 10: Proof for *Kim was eaten by Godzilla last night.*

$$\dfrac{\textbf{with}\ \ \lambda y\lambda P\lambda x\lambda e.[P(x)(e) \wedge animate(x) \wedge instrument(e) = y] : e \multimap (k \multimap ev \multimap t) \multimap k \multimap ev \multimap t \qquad \textbf{Excalibur}\ \ excalibur : e}{\lambda P\lambda x\lambda e.[P(x)(e) \wedge animate(x) \wedge instrument(e) = excalibur] : (k \multimap ev \multimap t) \multimap k \multimap ev \multimap t}$$

$$\dfrac{\textbf{tapped}\ \ tap' : s \multimap k \multimap ev \multimap t \qquad \textbf{Sandy}\ \ sandy : s}{tap'(sandy) : k \multimap ev \multimap t}$$

$$\dfrac{\lambda P\lambda x\lambda e.[P(x)(e) \wedge animate(x) \wedge instrument(e) = excalibur] : (k \multimap ev \multimap t) \multimap k \multimap ev \multimap t \qquad tap'(sandy) : k \multimap ev \multimap t}{\lambda x\lambda e.[tap'(sandy)(x)(e) \wedge animate(x) \wedge instrument(e) = excalibur] : k \multimap ev \multimap t}$$

$$\dfrac{\lambda x\lambda e.[tap'(sandy)(x)(e) \wedge animate(x) \wedge instrument(e) = excalibur] : k \multimap ev \multimap t \qquad \textbf{Kim}\ \ kim : k}{\lambda e.[tap'(sandy)(kim)(e) \wedge animate(kim) \wedge instrument(e) = excalibur] : ev \multimap t}$$

$$\dfrac{\textbf{PAST}\ \ \lambda P\exists e.[P(e) \wedge past(e)] : (ev \multimap t) \multimap t \qquad \lambda e.[tap'(sandy)(kim)(e) \wedge animate(kim) \wedge instrument(e) = excalibur] : ev \multimap t}{\exists e.[tap'(sandy)(kim)(e) \wedge animate(kim) \wedge instrument(e) = excalibur] : t}$$

$$\dfrac{\exists e.[tap'(sandy)(kim)(e) \wedge animate(kim) \wedge instrument(e) = excalibur] : t}{\exists e.[tap(e) \wedge agent(e) = kim \wedge patient(e) = sandy \wedge animate(kim) \wedge instrument(e) = excalibur \wedge past(e)] : t}\Rightarrow_\beta$$

Figure 11: Proof for *Kim tapped Sandy with Excalibur.*