An Improved Approach for Semantic Graph Composition with CCG

Austin Blodgett Nathan Schneider
Georgetown University, Department of Linguistics
{ajb341, nathan.schneider}@georgetown.edu

Abstract

This paper builds on previous work using Combinatory Categorial Grammar (CCG) to derive a transparent syntax-semantics interface for Abstract Meaning Representation (AMR) parsing. We define new semantics for the CCG combinators that is better suited to deriving AMR graphs. In particular, we define relation-wise alternatives for the application and composition combinators: these require that the two constituents being combined overlap in one AMR relation. We also provide a new semantics for type raising, which is necessary for certain constructions. Using these mechanisms, we suggest an analysis of eventive nouns, which present a challenge for deriving AMR graphs. Our theoretical analysis will facilitate future work on robust and transparent AMR parsing using CCG.

1 Introduction

At the heart of semantic parsing are two goals: the disambiguation of linguistic forms that can have multiple meanings, and the normalization of morphological and syntactic variation. Among many techniques for semantic parsing, one profitable direction exploits computational linguistic grammar formalisms that make explicit the correspondence between the linguistic form of a sentence and the semantics (e.g., broad-coverage logical forms, or database queries in a domain-specific query language). In particular, English semantic parsers using Combinatory Categorial Grammar (CCG; Steedman, 2000) have been quite successful thanks to the CCGBank resource (Hockenmaier and Steedman, 2007; Honnibal et al., 2010) and the broad-coverage statistical parsing models trained on it (e.g., Clark and Curran, 2004; Lewis et al., 2016; Clark et al., 2018).

The CCG formalism assumes that all language-specific grammatical information is stored in a lexicon: each word in the lexicon is associated with a structured syntactic **category** and a semantic form, such that the compositional potentials of the category and the semantics are isomorphic. A small universal set of **combinators** are responsible for assembling constituents into a full syntactic derivation; each combinator operates on adjacent constituents with appropriate categories to produce a new constituent and its compositional semantics, subject to constraints. A full grammar thus allows well-formed sentences to be transduced into semantic structures. The categories and combinators cooperate to license productive syntactic constructions like control and wh-questions, requiring the correct word order and producing the correct semantic dependencies. For example, consider the sentence "Who did John seem to forget to invite to attend?": the correct logical form—in propositional logic, something like $\mathit{seem}(\mathit{forget}(\mathit{John}_i, \mathit{invite}(\mathit{John}_i, \mathit{who}_j, \mathit{attend}(\mathit{who}_j))))$—is nontrivial, requiring a precise account of several constructions that conspire to produce long-range dependencies.

Whereas CCG traditionally uses some version of lambda calculus for its semantics, there has also been initial work using CCG to build parsers for Abstract Meaning Representation (AMR; Banarescu et al., 2013), a standard with which a large "sembank" of English sentences[1] has been manually annotated.[2] To

[1] See https://amr.isi.edu/download.html

[2] As originally defined, AMR is English-specific. However, a companion annotation standard, corpus, and parsers exist for Chinese (Xue et al., 2014; Li et al., 2016; Wang et al., 2018), and initial investigations have been made toward adapting AMR to several other languages (Xue et al., 2014; Migueles-Abraira et al., 2018; Anchiêta and Pardo, 2018).

date, dozens of publications[3] have used the corpus to train and evaluate semantic parsers—most using graph-based or transition-based parsing methods (e.g., Flanigan et al., 2014; Wang et al., 2016; Lyu and Titov, 2018) to transform the sentence string or syntactic parse into a semantic graph via a learned statistical model, without any explicit characterization of the syntax-semantics interface. There is good reason to apply CCG to the AMR parsing task: apart from transparency of the syntax-semantics interface, state-of-the-art AMR parsers are known to be weak at reentrancy (e.g., Lyu and Titov, 2018), which presumably can be partially attributed to syntactic reentrancy in control constructions, for example. Prior work applying CCG to AMR parsing has begun to address this, but some of the important mechanisms that make CCG a linguistically powerful and robust theory have yet to be incorporated into these approaches.

In this paper, we build on a core insight of previous work (e.g., Artzi et al., 2015; Beschke and Menzel, 2018) that AMR fragments can be directly represented as the semantics of CCG lexical entries. With appropriate definitions of the lexical items and combinatorial rules of CCG, the compositionality of CCG gives a derivation of a full AMR "for free". In other words, AMR parsing can be reduced to CCG parsing (plus some additional semantic disambiguation and postprocessing). On a practical level, this should allow us to take advantage of existing CCG datasets and parsing methods for AMR parsing. In addition, explicitly storing AMR fragments in the CCG lexicon would provide a level of interpretability not seen in most statistical AMR parsers: the transparent syntax-semantics interface would decouple errors in the grammar from errors in the parsing model.

As a prerequisite for building a CCG-based AMR parser, or inducing a broad-coverage grammar (CCG lexicon) from data, we consider in this paper the formal mechanisms that would be necessary to derive AMRs with linguistic robustness. In particular, we address a variety of challenging syntactic phenomena with respect to AMR, showing the semantic fragments, associated syntactic categories, and combinators that will facilitate parsing of constructions including control, wh-questions, relative clauses, case marking, nonconstituent coordination, eventive nouns, and light verbs. In so doing, we offer new semantics of combinators for semantic graphs beyond the proposals of previous work.

After an overview of related work (§2),[4] we introduce our formalism for AMR graph semantics in CCG (§3). §4 gives example derivations for well-known linguistic phenomena including control, complex coordination, and eventive nouns. §5 discusses some implications of our approach.

2 Related Work

AMR formalizes sentence meaning via a graph structure. The AMR for an English sentence is a directed acyclic graph that abstracts away from morphological and syntactic details such as word order, voice, definiteness, and morphology, focusing instead on lexical semantic predicates, roles, and relations. Semantic predicate-argument structures are based on the PropBank frame lexicon (Kingsbury and Palmer, 2002) and its frame-specific core argument roles (named *ARG0*, *ARG1*, etc.). AMR supplements these with its own inventory of noncore relations like :time and :purpose, and some specialized frames for the semantics of comparison, for example. Named entities are typed and linked to Wikipedia pages; dates and other values are normalized. Edges in the graph correspond to roles/relations, and nodes to predicate or non-predicate "concepts", which are lemmatized. Reentrancy is used for within-sentence coreference.

A limited amount of prior research has combined CCG and AMR. Artzi et al. (2015) and Misra and Artzi (2016) develop an AMR parser using CCG by reformulating AMR graphs as logical forms in lambda calculus. We opt here for an approach similar to that of Beschke and Menzel (2018), where AMR subgraphs with free variables are treated as the semantics in the CCG lexicon. This requires definitions of the combinators that operate directly on AMR subgraphs rather than lambda calculus expressions.

Beschke and Menzel (2018) situate their formalization within the literature on graph grammars. They formulate their approach in terms of the HR algebra (Courcelle and Engelfriet, 2012), which Koller (2015) had applied to AMR graphs (but not with CCG). In this formalism, graph fragments called s-graphs are assembled to derive full graphs. S-graphs are equivalent to the AMR subgraphs described in this paper.

[3] `https://nert-nlp.github.io/AMR-Bibliography/` is a categorized list of publications about or using AMR.

[4] Due to space constraints, we assume the reader is familiar with the basics of both CCG and AMR.

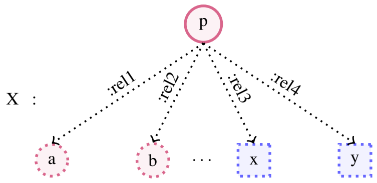


Figure 1: Basic shape of AMR subgraph: Free variables (square, blue) are represented with *x, y, z*, etc. AMR nodes (round, red) are represented with *a, b, c*, etc. Dots indicate that part of the graph may be present or not.

In particular, Beschke and Menzel define the semantics of CCG combinators in terms of HR-algebraic operations on s-graphs. They discuss a small set of combinators from Lewis and Steedman (2014) that includes forward and backward application and forward, backward, crossed, and generalized variants of composition. We introduce equivalent semantics for application and composition (§3.2), avoiding the conceptually heavy notation and formalism from the HR algebra. They also specify Conjunction and Identity combinators, which we adapt slightly to suit our needs, and a Punctuation combinator. More significantly, they treat unary operators such as type raising to have no effect on the semantics, whereas we will take another route for type raising (§3.4), and will introduce new, *relation-wise* versions of application and composition (§3.3). Finally, whereas Beschke and Menzel devote most of their paper to a lexicon induction algorithm and experiments, we focus on the linguistic motivation for our definition of the combinators, and leave the development of suitable lexicon induction techniques to future work.

A related graph formalism called *hyperedge replacement grammar* is also used in the AMR parsing literature (Jones et al., 2012; Chiang et al., 2013; Peng et al., 2015; Peng and Gildea, 2016; Björklund et al., 2016; Groschwitz et al., 2018). Hyperedge replacement grammars (Rozenberg, 1997) are a formal way of combining subgraphs to derive a larger graph, based on an extension of Context Free Grammars to graphs instead of strings. Readers may assume that the graph formalism described in this paper is a simplified hyperedge replacement grammar which only allows hyperedges of rank 1.

3 Graph Semantics

AMR is designed to represent semantics at the sentence level. For CCG lexical entries and combinators to parse AMR semantics, we need to formalize how AMR subgraphs can represent the semantics of individual words, and how combinators combine subgraphs to derive a full AMR. This section will formalize AMR subgraph semantics and CCG combinators for *function application*, *composition*, and *type raising*. Additionally, we propose new *relation-wise* variants of application and composition which are unique to graph semantics.

Each AMR subgraph contains nodes and edges from the resulting AMR as well as some nodes which correspond to free variables. The basic shape of an AMR subgraph appears in figure 1. Formally, an AMR subgraph is a tuple $\langle G, R, FV \rangle$, where G is a connected, labeled, directed acyclic graph; R is the root node in G; and FV is an ordered list of the nodes of G which are free and must be substituted by the end of the derivation. Though not shown in figure 1, the root of an AMR subgraph may be a free variable. Intuitively, a subgraph with at least one free variable corresponds to a function, and a subgraph with no free variables corresponds to a constant.

Textual notation. Taking inspiration from the PENMAN notation used for AMR, we use the notation $(a$:rel1 $(\boxed{2}$:rel2 $\boxed{1}))$ to denote an AMR subgraph rooted at a constant a, with a :rel1 edge to a free variable, $\boxed{2}$, which in turn has a child free variable, $\boxed{1}$.

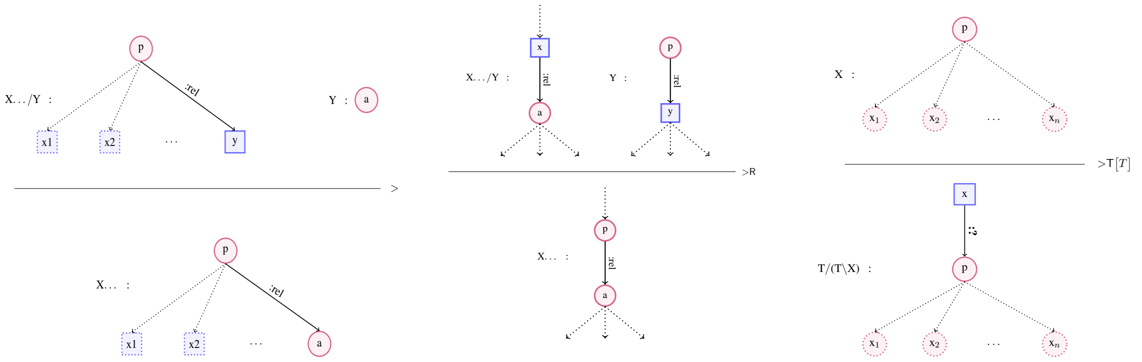
Table 1 shows the formulation of graph semantics for all the combinators described below. The formulas are schematic with attention paid to the resulting order of free variables, which semantically distinguishes application from composition. Another combinator in CCG, crossing composition, has the same semantics as regular composition. Semantics for the substitution combinator is left to future work.

3.1 Syntax-Semantics Isomorphism

A core property of CCG is that it provides transparency in the syntax-semantics interface: both syntactic categories and semantic forms are defined as functions permitting a compositional derivation of the

combinator	*function (left/right)*	*arg. (right/left)*	*result*	*FV ordering*
Binary				
Application	$\ldots_1\ \boxed{1}\ \ldots_2$	$\mathrm{a}\ \ldots_3$	$\ldots_1\ \mathrm{a}\ \ldots_2\ \ldots_3$	$\boxed{2}, \ldots, \fcolorbox{black}{lightgray}{1}, \ldots$
Composition (B, B^2)	$\ldots_1\ \boxed{1}\ \ldots_2$	$\mathrm{a}\ \ldots_3$	$\ldots_1\ \mathrm{a}\ \ldots_2\ \ldots_3$	$\fcolorbox{black}{lightgray}{1}, \ldots, \boxed{2}, \ldots$
Relation-wise Application (R)	$\ldots_1\ \boxed{1}\ \ \mathrm{:rel}_x\ \ \mathrm{b}\ \ldots_2$	$\mathrm{a}\ \ \mathrm{:rel}_x\ \ \fcolorbox{black}{lightgray}{1}\ \ldots_3$	$\ldots_1\ \mathrm{a}\ \ \mathrm{:rel}_x\ \ \mathrm{b}\ \ldots_2\ \ldots_3$	$\boxed{2}, \ldots, \fcolorbox{black}{lightgray}{2}, \ldots$
Relation-wise Composition (RB)	$\ldots_1\ \boxed{1}\ \ \mathrm{:rel}_x\ \ \mathrm{b}\ \ldots_2$	$\mathrm{a}\ \ \mathrm{:rel}_x\ \ \fcolorbox{black}{lightgray}{2}\ \ldots_3$	$\ldots_1\ \mathrm{a}\ \ \mathrm{:rel}_x\ \ \mathrm{b}\ \ldots_2\ \ldots_3$	$\fcolorbox{black}{lightgray}{1}, \fcolorbox{black}{lightgray}{3}, \ldots, \boxed{2}, \ldots$
. . . Second-order (RB^2)	$\ldots_1\ \boxed{1}\ \ \mathrm{:rel}_x\ \ \mathrm{b}\ \ldots_2$	$\mathrm{a}\ \ \mathrm{:rel}_x\ \ \fcolorbox{black}{lightgray}{3}\ \ldots_3$	$\ldots_1\ \mathrm{a}\ \ \mathrm{:rel}_x\ \ \mathrm{b}\ \ldots_2\ \ldots_3$	$\fcolorbox{black}{lightgray}{1}, \fcolorbox{black}{lightgray}{2}, \fcolorbox{black}{lightgray}{4}, \ldots, \boxed{2}, \ldots$
Unary				
Type Raising (T)	$\mathrm{a}\ \ldots_1$		$\boxed{1}\ \ \mathbf{:?}\ \ \mathrm{a}\ \ldots_1$	$\boxed{1}, \fcolorbox{black}{lightgray}{1}, \ldots$
N-ary ($\leq$1 FV per operand)				
Conjunction (&)	x	$\mathrm{a}\ \ldots_1, \mathrm{b}\ \ldots_2, \ldots$	$\mathrm{x}\ \mathrm{:op1}\ \mathrm{a}\ \ldots_1\ \mathrm{:op2}\ \mathrm{b}\ \ldots_2\ \ldots$	$\boxed{1}$

Table 1: Formal semantic rules for AMR combinators. Boxed numbers stand for free variables (FVs) in the semantics of each of the constituents being combined: $\boxed{1}$ stands for the lowest indexed FV in the function (head) constituent, and $\fcolorbox{black}{lightgray}{1}$ for the lowest indexed FV in the argument constituent, if any. Ellipses $\ldots_n$ denote optional dominating structure (if preceding) and optional dominated structure (if following). Any FVs in these optional structures are preserved in the result, in the order given in the last column. For relation-wise combinators, the function constituent's relation may also be **:?**. Crossing composition (B$_\times$) and its variants behave semantically like their non-crossing counterparts. Not shown: exceptions to application and composition for the identity function (ID), discussed in §4.



(a) Function Application (b) Relation-wise Application (c) Type Raising

Figure 2: Combinators illustrated in terms of semantic graph structure. The semantics of composition differs from application only in ordering of free variables (not shown).

sentence. The syntactic category determines which constituents may be constructed and in what word order. In the semantics, the word order (direction of the slashes) is irrelevant, but the functional structure—the arity and the order in which arguments are to be applied—must match in order for the semantics to remain well-formed as the sentence is derived based on the syntactic categories and combinatorial rules.

In other words, the functional structure of the category must be isomorphic to the functional structure of the semantics. For example, a hypothetical CCG category V\W/X/(Y/Z) would naturally correspond to a ternary function whose first argument, Y/Z, is itself a unary function.

This brings us to the following principle:

Principle of Functional Isomorphism. The semantics of a word or constituent cannot have higher arity than the CCG category calls for, and every functional category must take at least one semantic argument. For instance, a word or constituent with category PP/NP must have exactly 1 semantic argument; and the VP adjunct category (S\NP)\(S\NP) a.k.a. S\NP\(S\NP) can be interpreted as having 1 or 2 semantic arguments.

Without proving it formally, we remark that this helps ensure that syntactic well-formedness according to the categories will guarantee semantic well-formedness, with no attempt to apply something that is not expecting any arguments, and no free variables remaining in the semantics at the end of a sentence derivation. (An edge case where this guarantee might not hold is noted in fn. 6.)

3.2 Function Application and Composition

In **Function Application** of AMR subgraphs, a free variable (blue) can be filled by the root of another AMR subgraph. The case of right function application is shown in figure 2a. Function application can only substitute the first free variable in FV corresponding to the rightmost syntactic argument.

While application and composition always differ syntactically, from a graph semantics point of view, composition turns out to be the same as function application, where the root of one subgraph is substituted for a free variable in another subgraph. The difference between application and composition is captured in the resulting order of free variables. In the case of composition, the argument's free variables are placed first on the free variable stack followed by the function's free variables. This allows free variables in the AMR subgraph to consistently match syntactic arguments in the CCG category. This is a difference between composition in this work and in Beschke and Menzel's (2018) work, where the semantics of application and composition is the same.

3.3 Relation-wise Application and Composition

When deriving a constituent, there are situations where it is desirable to have a semantic edge that is shared between the two constituents being combined. For example, we specify the following lexical entry for the control verb "decide", indexing arguments in the category with subscripts for clarity: $\mathrm{S}_b\backslash\mathrm{NP}_2/(\mathrm{S}_{to}\backslash\mathrm{NP})_1$: decide-01 :ARG0 $\boxed{2}$:ARG1 ($\boxed{1}$ **:ARG0** $\boxed{2}$). Unlike a simple verb, "decide" selects for an embedded clause and controls its subject, coindexing it with the matrix subject. This is indicated in the semantics with the bolded :ARG0 edge, which needs to unify with the :ARG0 edge of the embedded predicate. Thus the constituent "you decide to eat yesterday" in figure 7 is formed by merging the :ARG0 edge expected by "decide" and the :ARG0 edge expected by "eat" so that they may later be filled by the same node, you. Note that the number of semantic free variables respects the functional structure of the category (§3.1). To facilitate this, we define novel **relation-wise** variants of the application and composition combinators that expect an edge in common (call it the **shared edge**). Apart from control, relation-wise combinators are also useful for derivations with type raising and various interesting syntactic constructions.

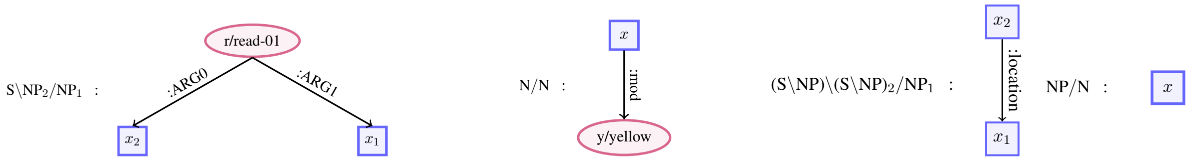
The schematic graph structures serving as inputs and outputs for relation-wise combinators are shown in figure 2b, and the full definition is given in table 1. Notably, the function constituent has its lowest-indexed free variable at the *source* of the shared edge, and the argument constituent has a free variable at the *target* of the shared edge (the variable's index depending on the kind of application or composition). In the result, each free variable unifies with the node or variable at the same side of the edge in the other constituent. Other material attached to the shared edge in either constituent will be preserved in the result.

The regular vs. relation-wise distinction applies only to the *semantics*; syntactically, relation-wise application (composition) is just like regular application (composition). During parsing, relation-wise combinators apply if and only if the two constituents being combined share a common relation with the appropriate free variables; otherwise, the non–relation-wise version of the combinator is used.

Relation-wise Composition (RB) differs from **Relation-wise Application** (R) in the index of the argument's free variable being unified and in the resulting order of free variables. Just as regular composition can be used to adjust the order that constituents are normally combined and "save an argument for later", relation-wise composition does this with respect to a common edge. Examples of both relation-wise and non–relation-wise composition appear in figure 7.

3.4 Type Raising

In CCG, **Type Raising** (T) converts an argument into a function. For example, the nominative case of the pronoun "I" can be coded in the syntactic category by making it a function that expects a verb phrase on the right and returns a sentence, thus preventing "I" from serving as an object. For our framework to support type raising, we need an appropriate semantic conversion that respects the functional structure of the category—thus, the type-raised semantics must take an argument. However, as type raising can be applied to different types of arguments, we do not know a priori which relation label to produce.



(a) "read"; (r/read-01 :ARG0 $\boxed{2}$:ARG1 $\boxed{1}$) (b) "yellow"; ($\boxed{1}$:mod y/yellow) (c) "at"; ($\boxed{2}$:location $\boxed{1}$) (d) "the"; ID

Figure 3: Linguistic examples as AMR subgraphs: (a) transitive verb, (b) adjective, (c) preposition (in VP adjunct), (d) determiner (identity semantics).

Therefore, we introduce the notion of an **underspecified edge**, denoted **:?**. The type-raised structure has a free variable at the source of the underspecified edge, with the original subgraph at the target, as shown in figure 2c. For example, see "John" and "Mary" in figure 5, where type raising is necessary to support subject+verb constituents for coordination. The type-raised constituent must eventually be the input to a relation-wise combinator, which will specify the label on the edge.

Note that in this strategy of representing type raising, the isomorphism between functions in semantics and syntactic category is maintained. This fits with CCG's philosophy of a transparent syntax-semantics interface (§3.1). By contrast, Beschke and Menzel's (2018) strategy was to leave the result of type raising semantically unchanged, creating a mismatch between the syntax and the semantics.

4 Linguistic Examples

This section explains the use of the combinators discussed in §3 for particular linguistic constructions.
Transitive and Intransitive Verbs. Figure 3a shows the semantics for a transitive verb. Since "read" has more than one semantic argument, the order of free variables matters: $\boxed{1}$, the first free variable, must correspond to NP_1, the rightmost syntactic argument in the category.
Adjectives. Figure 3b shows the semantics for an adjective. Note that, unlike in the examples above, the root of this subgraph is a free variable, since the root of this subgraph is what will be filled in. Ordinary adverbs have similar semantics.
Prepositional Phrases (Adjunct). Figure 3c shows semantics for the locative preposition "at". To derive a prepositional phrase, assume available constituents "at": ($\boxed{2}$:location $\boxed{1}$) and "the library": (l/library), which may be combined by application.
Null Semantics: Articles, etc. Some linguistic features, including tense and definite/indefinite articles, are not represented in AMR. For CCG derivations to deal with these elements, there will need to be a semantic representation which allows them to be "syntactic sugar", affecting the syntactic category but adding nothing to the semantics in the derivation. We call this the **identity function**, following Beschke and Menzel (2018), and notate it as ID. More precisely, if a constituent a has ID as its semantics, then a, when combined with another constituent b via application or composition (either as the function or as the argument), will produce b's semantics for the resulting constituent.

Figure 4 shows the use of application (and identity application) combinators to derive a simple sentence. Figure 5 demonstrates type raising, relation-wise composition, and conjunction as tools to derive a sentence with complex coordination.
Passives, Control, and Wh-questions. Figures 6 and 7 show CCG derivations with AMR semantics for three well-known linguistic phenomena in English: passives, control, and wh-questions. In a passive

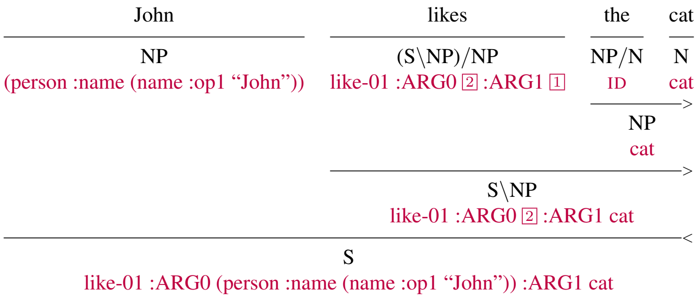

John
likes
the
cat
NP
(person :name (name :op1 "John"))
(S\NP)/NP
like-01 :ARG0 2 :ARG1 1
NP/N
ID
N
cat
>
NP
cat
>
S\NP
like-01 :ARG0 2 :ARG1 cat
<
S
like-01 :ARG0 (person :name (name :op1 "John")) :ARG1 cat

Figure 4: **application and identity**: "John likes the cat"

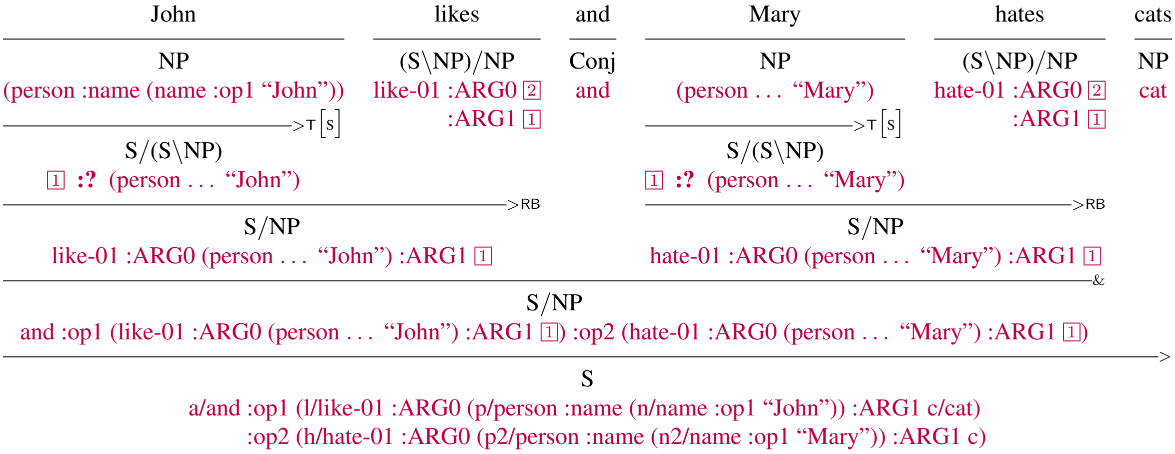

John likes and Mary hates cats
NP (person :name (name :op1 "John"))
(S\NP)/NP like-01 :ARG0 2 :ARG1 1
Conj and
NP (person … "Mary")
(S\NP)/NP hate-01 :ARG0 2 :ARG1 1
NP cat
>T [s]
S/(S\NP) 1 :? (person … "John")
S/(S\NP) 1 :? (person … "Mary")
>RB
S/NP like-01 :ARG0 (person … "John") :ARG1 1
S/NP hate-01 :ARG0 (person … "Mary") :ARG1 1
&
S/NP and :op1 (like-01 :ARG0 (person … "John") :ARG1 1) :op2 (hate-01 :ARG0 (person … "Mary") :ARG1 1)
>
S
a/and :op1 (l/like-01 :ARG0 (p/person :name (n/name :op1 "John")) :ARG1 c/cat)
:op2 (h/hate-01 :ARG0 (p2/person :name (n2/name :op1 "Mary")) :ARG1 c)

Figure 5: **complex coordination and type raising**: "John likes and Mary hates cats"

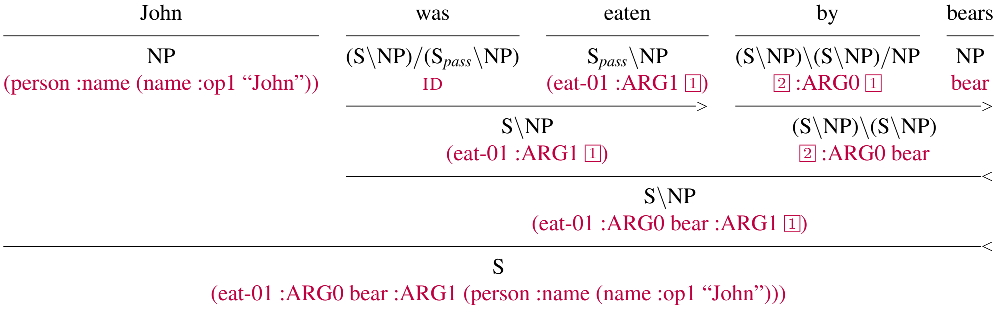

John was eaten by bears
NP (person :name (name :op1 "John"))
(S\NP)/(S_pass\NP) ID
S_pass\NP (eat-01 :ARG1 1)
(S\NP)\(S\NP)/NP 2 :ARG0 1
NP bear
>
S\NP (eat-01 :ARG1 1)
(S\NP)\(S\NP) 2 :ARG0 bear
<
S\NP (eat-01 :ARG0 bear :ARG1 1)
<
S (eat-01 :ARG0 bear :ARG1 (person :name (name :op1 "John")))

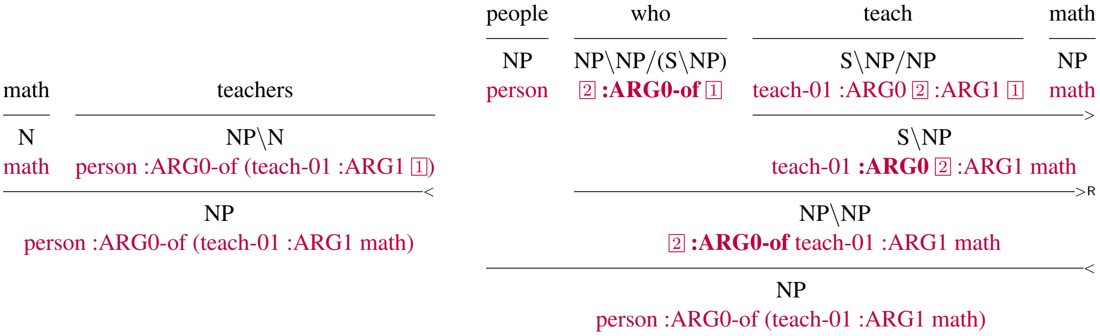
Figure 6: **passive**: "John was eaten by bears"

construction, a semantically core argument may be added by a syntactically optional adjunct phrase as in figure 6. Note that in this semantic representation, only syntactically required arguments are represented in a predicate's semantics, and so the passive verb *eaten* does not include an :ARG0 edge.

Figure 7 shows both control and wh-question formation. Control is an important problem for graph semantics as it requires representing the subject (here *you*) as the agent of two predicates (see §3.3). Wh-questions are another complex and difficult phenomenon that is handled by CCG derivation. Additionally, figure 7 gives examples of both types of composition: relation-wise and non–relation-wise.

4.1 Inverse Core Roles and Relative Clauses

AMR provides notation for *inverse roles* that reverse the usual ordering of a relation. These are indicated with the -of suffix: (a :rel-of b) is equivalent to (b :rel a). This ensures that the graph can be constructed with a single root, and provides a convenient mechanism for expressing derived nominals and relative clauses. For instance, the noun phrases "teacher" and "a person who teaches" both receive the AMR (person :ARG0-of teach-01). If the subject matter is expressed, that is slotted into the :ARG1 of teach-01. This can be handled by treating "teachers" as a predicate of sorts, as seen in the derivation below.


math teachers
N math
NP\N person :ARG0-of (teach-01 :ARG1 1)
<
NP person :ARG0-of (teach-01 :ARG1 math)
people who teach math
NP person
NP\NP/(S\NP) 2 :ARG0-of 1
S\NP/NP teach-01 :ARG0 2 :ARG1 1
NP math
>
S\NP teach-01 :ARG0 2 :ARG1 math
>R
NP\NP 2 :ARG0-of teach-01 :ARG1 math
<
NP person :ARG0-of (teach-01 :ARG1 math)

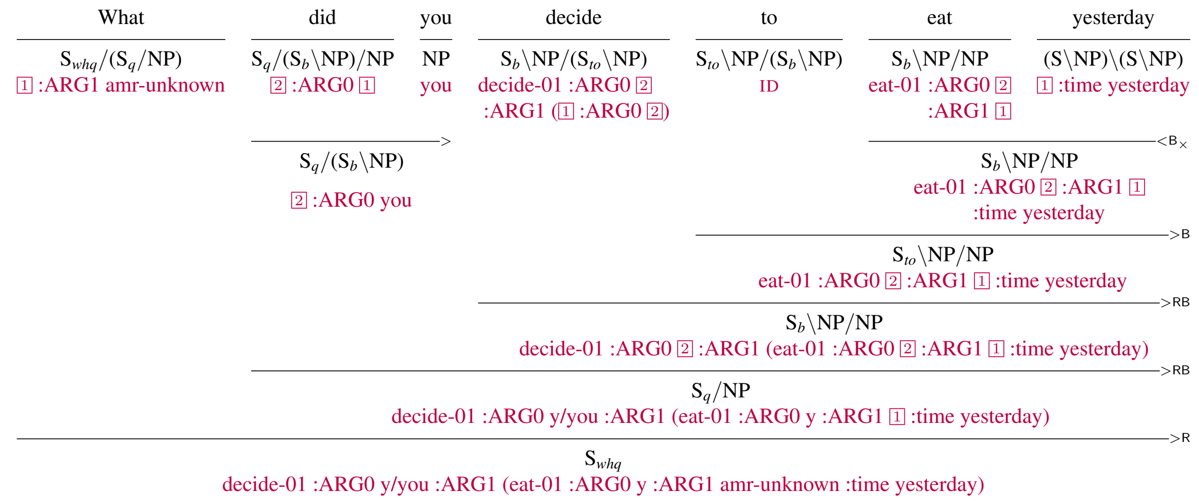

What
did
you
decide
to
eat
yesterday
S_whq/(S_q/NP)
1 :ARG1 amr-unknown
S_q/(S_b\NP)/NP
2 :ARG0 1
NP
you
S_b\NP/(S_to\NP)
decide-01 :ARG0 2 :ARG1 (1 :ARG0 2)
S_to\NP/(S_b\NP)
ID
S_b\NP/NP
eat-01 :ARG0 2 :ARG1 1
(S\NP)\(S\NP)
1 :time yesterday
>
S_q/(S_b\NP)
2 :ARG0 you
<B×
S_b\NP/NP
eat-01 :ARG0 2 :ARG1 1 :time yesterday
>B
S_to\NP/NP
eat-01 :ARG0 2 :ARG1 1 :time yesterday
>RB
S_b\NP/NP
decide-01 :ARG0 2 :ARG1 (eat-01 :ARG0 2 :ARG1 1 :time yesterday)
>RB
S_q/NP
decide-01 :ARG0 y/you :ARG1 (eat-01 :ARG0 y :ARG1 1 :time yesterday)
>R
S_whq
decide-01 :ARG0 y/you :ARG1 (eat-01 :ARG0 y :ARG1 amr-unknown :time yesterday)

Figure 7: **wh-question and control, relation-wise and non–relation-wise composition**: "What did you decide to eat yesterday?" $\mathsf{B}_\times$ stands for *crossing composition*, which has the same semantics as composition.

Also illustrated is the relative clause paraphrase, "people who teach math". Here, the relativizer "who" needs to fill the appropriate role of the verbal predicate with its noun head "people". An inverse role is produced so that person, rather than teach-01, will be the root of the resulting subgraph. The relation-wise application combinator must therefore be aware of inverses: it must match the :ARG0-of with the :ARG0 edge in the operand and effectively merge the two relations. Alternatively, the phrase could be parsed by first relation-wise composing "who" with "teach", which requires similar handling of the inverse role, and then attaching "math" by application.

4.2 Eventive Nouns and PP Complements

This section will describe an approach to the semantics of eventive nouns like "decision", and in the process will illustrate our treatment of prepositional phrase complements (as opposed to adjuncts: beginning of §4), which in CCG are traditionally given the category PP.

In English, many eventive nouns can be linked to semantic arguments via prepositional phrases, possessives, and light verb constructions, as shown in table 2. AMR uses a canonical form with a predicate (typically based on a verbal paraphrase), treating *John decided*, *John's decision*, and *John made a/his decision* as semantically equivalent. Despite some work on integrating event nominals and multiword expressions into CCG (Constable and Curran, 2009; Honnibal et al., 2010; de Lhoneux, 2014), we are not aware of any CCG analyses of **light verb constructions**, which have been studied computationally in other frameworks (e.g., Baldwin and Kim, 2010; Bonial et al., 2014; Ramisch et al., 2018), that gives them semantics equivalent to a content verb paraphrase. We offer such an analysis based on three principles:

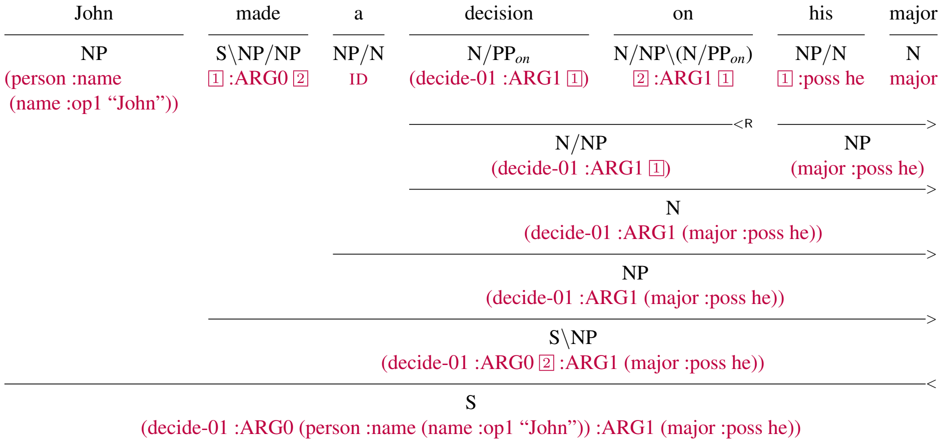
1. The event frame is in the semantics of the eventive noun or verb.
2. For any syntactic argument of a noun or verb, the corresponding edge (and free variable) is in the semantics of the noun or verb.
3. Any function word (light verb, *'s*, preposition, or infinitival *to*) that links the eventive noun to its semantic argument has an associated edge (and free variables) in its semantics.

Note that when a verb or noun takes a PP complement, principles 2 and 3 force both the verb or noun and the preposition to hold the same edge in their semantics. This is compatible with relation-wise combinators as described in §3.3. The result is a nice analysis where both the eventive noun or verb and its complement preposition signal *patientness*.

With this analysis, the associated light verbs given in table 2 ("make", "pay", etc.) as well as possessive *'s* take the semantics $\boxed{1}$:ARG0 $\boxed{2}$, and associated prepositions take the semantics $\boxed{2}$:ARG1 $\boxed{1}$. In other words, for each eventive noun, either a special light verb or a possessive contributes the agentive semantic

light verb construction	possessive form	AMR predicate
make a **decision** [about/on]	my **decision** [about/on]	decide-01
pay **attention** [to]	my **attention** [to]	attend-02
make an **attempt** [to]	my **attempt** [to]	attempt-01
take a **nap**	my **nap**	nap-01
take a **picture** [of]	— ("my picture" is not eventive)	photograph-01 *(suggested)*

Table 2: English eventive nouns shown with a light verb or possessive; words in square brackets mark additional semantic arguments. (In the AMR corpus, "take pictures" is actually treated superficially with take-01 :ARG1 picture, but we suggest photograph-01 instead.)



John
made
a
decision
on
his
major
NP
(person :name (name :op1 "John"))
S\NP/NP
1 :ARG0 2
NP/N
ID
N/PP_on
(decide-01 :ARG1 1)
N/NP\(N/PP_on)
2 :ARG1 1
NP/N
1 :poss he
N
major
<R
>
N/NP
(decide-01 :ARG1 1)
NP
(major :poss he)
>
N
(decide-01 :ARG1 (major :poss he))
>
NP
(decide-01 :ARG1 (major :poss he))
>
S\NP
(decide-01 :ARG0 2 :ARG1 (major :poss he))
<
S
(decide-01 :ARG0 (person :name (name :op1 "John")) :ARG1 (major :poss he))

Figure 8: **light verb construction**: "John made a decision on his major"

relation—and (if present) a special preposition or infinitive *to* may contribute the patient semantic relation—thus allowing derivation of the same AMR regardless of form.

Figure 8 shows the derivation for "decision" in its light verb construction form. The preposition "on" redundantly represents the :ARG1 edge, and is merged with "decision" by relation-wise application.[5] The light verb "made" specifies the :ARG0 edge.

5 Discussion

Unlike many semantic formalisms, AMR does not specify a 'compositional story': annotations do not include any sort of syntactic derivation, or even gold alignments between semantic units and words in the sentence. This presents a challenge for AMR parsing, which in practice relies on various forms of automatic or latent alignments (see Szubert et al., 2018). Above, we have presented an analysis that lays the foundation for a linguistically principled treatment of CCG-to-AMR parsing that meets a variety of challenges in the syntax-semantics interface, and does so in a transparent way so that parsing errors can be diagnosed. We believe the approach is reasonably intuitive, flowing naturally from CCG syntax, AMR semantics, and the notion of free variables in subgraphs, without the additional need for complicated lambda calculus notation or a highly general graph grammar formalism.

To realize this vision in practice, an approach is needed to build a CCG parser enriched with graph semantics for deriving AMRs. We anticipate that existing CCG parsing frameworks can be adapted—for example, by developing an alignment algorithm to induce the semantics for lexical entries from the AMR corpus, and running an off-the-shelf parser like EasySRL (Lewis et al., 2015) at training and test time for the syntactic side of the derivation. This approach would take advantage of the fact that our analysis assumes the ordinary CCG syntax for obtaining the compositional structure of the derivation. The only additional steps would be a) disambiguating the semantics of lexical entries in the derivation, and b) applying the semantics of the combinators as specified in table 1. For each use of application or

[5] The category N/NP\(N/PP$_{on}$) for "on" is suggested by Mark Steedman's analysis of English prepositions as particles (personal communication) and also maintains the Principle of Functional Isomorphism of §3.1.

composition, the semantic parser would check whether the conditions for relation-wise combination hold, and otherwise apply the ordinary version of the combinator.[6]

Because AMRs are annotated by humans for raw sentences, rather than on top of a syntactic parse, we cannot expect a parser to elegantly handle the full construction of all AMRs according to compositional rules. Several components of AMR parsing are not part of CCG parsing and will have to be performed as postprocessing steps. These components include named entity recognition, time expression parsing, coreference resolution, and wikification, all of which need to be performed after (or before) CCG parsing. Additionally, there is a risk that a CCG lexicon may 'overgenerate', producing invalid parses, and additional checking—either in the combinators, or as postprocessing or reranking—may be warranted.

We are aware of certain phenomena where the approach described above would be unable to fully match the conventions of AMR in the CCG-derived semantics. The analysis presented for **coordination** (with the conjunction combinator: see figure 5) would address only one of the ways it can be expressed in AMR, with a concept like and or or and operands. In other cases, coordinated modifiers are treated as sister relations in the AMR, with no explicit concept for the conjunction. Even when the conjunction is explicit in the AMR, it may be placed at a different level in the gold and CCG-derived AMRs: e.g., when two purpose adjuncts are coordinated, the derivation will result in semantic conjunction over the predicate rather than a conjunction under the :purpose relation. In sentences where a semantic predicate is **duplicated** in the AMR with different participants, e.g. due to right node raising, a copy mechanism would be needed to avoid spurious reentrancy. The treatment of **modal auxiliaries** as above the main event predicate in the AMR will be problematic for the CCG derivation when there is a preposed adjunct (as in "*Tomorrow*, John may eat rice") because the modifier will semantically attach under the root of the semantics of the rest of the clause (possible-01 from "may") rather than the main event predicate eat-01. Full derivations for these problem cases, as well as examples of purpose clauses, raising, and subject and object control, are given in appendix A. We will explore whether such limitations can be addressed via postprocessing of the parse, or whether additional expressive power in the combinators is necessary.

Finally, as pointed out by Bender et al. (2015), AMR annotations sometimes go beyond the compositional 'sentence meaning' and incorporate elements of 'speaker meaning', though an empirical study of AMR data found the rate of noncompositional structures to be relatively low (Szubert et al., 2018). Beschke and Menzel (2018) give interesting examples of AMR fragments that would be difficult to derive compositionally, e.g., "settled on Indianapolis for its board meeting", where the AMR attaches Indianapolis as the location of the meeting and the meeting as the thing that was settled on (reflecting the inference *settle on* LOCATION *for* ACTIVITY ⇒ *settle on* [ACTIVITY *at* LOCATION]).

6 Conclusion

We have given the linguistic motivation for a particular method of deriving AMR semantic graphs using CCG. Our specification of AMR subgraphs and CCG combinators ensures a tight correspondence between syntax and semantics, which we have illustrated for a variety of linguistic constructions (including light verb construction semantics, which to the best of our knowledge has not previously been explored for CCG). Future empirical work can make use of this framework to induce CCG lexicons for AMR parsing.

[6] We have considered an alternative analysis where underspecified **:?** edges would be used not only for type raising, but for all case-marked pronouns, prepositions marking syntactic arguments, and other constructions where a word's syntactic category involves an argument to a separate predicate. Thus, only the predicate would be allowed to specify semantic roles for its syntactic arguments. Relation-wise combinators would then require that the shared edge would be underspecified in the function constituent. The rationale would be that this avoids redundant specification of core roles like :ARG0 and :ARG1 in the lexical entries—e.g. in figure 7, the :ARG1 for "What", the :ARG0 for "did", and the second :ARG0 for "decide" would all be replaced with **:?**. After all, constructions like wh-questions, control, and case target syntactic relations (subject/object), which are merely *correlated* with semantic roles. And as pointed out by a reviewer, under the current approach, a wrong choice of semantic role for a cased pronoun's semantics could result in the use of a regular combinator rather than a relation-wise combinator, leaving a free variable in the predicate unsatisfied and essentially breaking the syntax-semantics isomorphism. An argument in favor of the current policy is that prepositions can contain information about roles to a certain extent, and redundant specification of semantic roles may actually be helpful when confronted with a noisy parser and lexicon. We leave this open as an empirical question for parsing research.

Acknowledgments

We want to thank Paul Portner, Adam Lopez, members of the NERT lab at Georgetown, and anonymous reviewers for their helpful feedback on this research, as well as Matthew Honnibal, Siva Reddy, and Mark Steedman for early discussions about light verbs in CCG.

References

Anchiêta, R. T. and T. A. S. Pardo (2018, May). Towards AMR-BR: A SemBank for Brazilian Portuguese language. In N. Calzolari, K. Choukri, C. Cieri, T. Declerck, S. Goggi, K. Hasida, H. Isahara, B. Maegaard, J. Mariani, H. Mazo, A. Moreno, J. Odijk, S. Piperidis, and T. Tokunaga (Eds.), *Proc. of LREC*, Miyazaki, Japan, pp. 974–979.

Artzi, Y., K. Lee, and L. Zettlemoyer (2015, September). Broad-coverage CCG semantic parsing with AMR. In *Proc. of EMNLP*, Lisbon, Portugal, pp. 1699–1710.

Baldwin, T. and S. N. Kim (2010). Multiword expressions. In N. Indurkhya and F. J. Damerau (Eds.), *Handbook of Natural Language Processing, Second Edition*, pp. 267–292. Boca Raton, FL: CRC Press, Taylor and Francis Group.

Banarescu, L., C. Bonial, S. Cai, M. Georgescu, K. Griffitt, U. Hermjakob, K. Knight, P. Koehn, M. Palmer, and N. Schneider (2013, August). Abstract Meaning Representation for sembanking. In *Proc. of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, Sofia, Bulgaria, pp. 178–186.

Bender, E. M., D. Flickinger, S. Oepen, W. Packard, and A. Copestake (2015, April). Layers of interpretation: on grammar and compositionality. In *Proc. of IWCS*, London, UK, pp. 239–249.

Beschke, S. and W. Menzel (2018, June). Graph Algebraic Combinatory Categorial Grammar. In *Proc. of *SEM*, New Orleans, Louisiana, pp. 54–64.

Björklund, H., F. Drewes, and P. Ericson (2016). Between a rock and a hard place – Uniform parsing for hyperedge replacement DAG grammars. In A. Dediu, J. Janoušek, C. Martín-Vide, and B. Truthe (Eds.), *Language and Automata Theory and Applications*, Lecture Notes in Computer Science, pp. 521–532.

Bonial, C., M. Green, J. Preciado, and M. Palmer (2014, April). An approach to ‘take’ multi-word expressions. In *Proc. of the 10th Workshop on Multiword Expressions*, Gothenburg, Sweden, pp. 94–98.

Chiang, D., J. Andreas, D. Bauer, K. M. Hermann, B. Jones, and K. Knight (2013, August). Parsing graphs with hyperedge replacement grammars. In *Proc. of ACL*, Sofia, Bulgaria, pp. 924–932.

Clark, K., M. Luong, C. D. Manning, and Q. Le (2018, November). Semi-supervised sequence modeling with cross-view training. In *Proc. of EMNLP*, Brussels, Belgium, pp. 1914–1925.

Clark, S. and J. R. Curran (2004, July). Parsing the WSJ using CCG and log-linear models. In *Proc. of ACL*, Barcelona, Spain, pp. 103–110.

Constable, J. and J. Curran (2009). Integrating verb-particle constructions into CCG parsing. In *Proc. of the Australasian Language Technology Association Workshop 2009*, Sydney, Australia, pp. 114–118.

Courcelle, B. and J. Engelfriet (2012, June). *Graph Structure and Monadic Second-Order Logic: A Language-Theoretic Approach*. Cambridge University Press.

de Lhoneux, M. (2014, August). CCG parsing and multiword expressions. `http://arxiv.org/abs/1505.04420`.

Flanigan, J., S. Thomson, J. Carbonell, C. Dyer, and N. A. Smith (2014, June). A discriminative graph-based parser for the Abstract Meaning Representation. In *Proc. of ACL*, Baltimore, Maryland, USA, pp. 1426–1436.

Groschwitz, J., M. Lindemann, M. Fowlie, M. Johnson, and A. Koller (2018, July). AMR dependency parsing with a typed semantic algebra. In *Proc. of ACL*, Melbourne, Australia, pp. 1831–1841.

Hockenmaier, J. and M. Steedman (2007, August). CCGbank: A corpus of CCG derivations and dependency structures extracted from the Penn Treebank. *Computational Linguistics 33*(3), 355–396.

Honnibal, M., J. R. Curran, and J. Bos (2010, July). Rebanking CCGbank for improved NP interpretation. In *Proc. of ACL*, Uppsala, Sweden, pp. 207–215.

Jones, B., J. Andreas, D. Bauer, K. M. Hermann, and K. Knight (2012, December). Semantics-based machine translation with hyperedge replacement grammars. In *Proc. of COLING 2012*, Mumbai, India, pp. 1359–1376.

Kingsbury, P. and M. Palmer (2002, May). From TreeBank to PropBank. In *Proc. of LREC*, Las Palmas, Canary Islands, pp. 1989–1993.

Koller, A. (2015, April). Semantic construction with graph grammars. In *Proc. of IWCS*, London, UK, pp. 228–238.

Lewis, M., L. He, and L. Zettlemoyer (2015, September). Joint A* CCG parsing and semantic role labelling. In *Proc. of EMNLP*, Lisbon, Portugal, pp. 1444–1454.

Lewis, M., K. Lee, and L. Zettlemoyer (2016, June). LSTM CCG Parsing. In *Proc. of NAACL-HLT*, San Diego, California, USA, pp. 221–231.

Lewis, M. and M. Steedman (2014, October). A* CCG parsing with a supertag-factored model. In *Proc. of EMNLP*, Doha, Qatar, pp. 990–1000.

Li, B., Y. Wen, L. Bu, W. Qu, and N. Xue (2016, August). Annotating The Little Prince with Chinese AMRs. In *Proc. of LAW X – the 10th Linguistic Annotation Workshop*, Berlin, Germany, pp. 7–15.

Lyu, C. and I. Titov (2018, July). AMR parsing as graph prediction with latent alignment. In *Proc. of ACL*, Melbourne, Australia, pp. 397–407.

Migueles-Abraira, N., R. Agerri, and A. D. d. Ilarraza (2018, May). Annotating Abstract Meaning Representations for Spanish. In N. Calzolari, K. Choukri, C. Cieri, T. Declerck, S. Goggi, K. Hasida, H. Isahara, B. Maegaard, J. Mariani, H. Mazo, A. Moreno, J. Odijk, S. Piperidis, and T. Tokunaga (Eds.), *Proc. of LREC*, Miyazaki, Japan, pp. 3074–3078.

Misra, D. K. and Y. Artzi (2016, November). Neural shift-reduce CCG semantic parsing. In *Proc. of EMNLP*, Austin, Texas, pp. 1775–1786.

Peng, X. and D. Gildea (2016, June). UofR at SemEval-2016 Task 8: Learning Synchronous Hyperedge Replacement Grammar for AMR parsing. In *Proc. of SemEval*, San Diego, California, USA, pp. 1185–1189.

Peng, X., L. Song, and D. Gildea (2015, July). A Synchronous Hyperedge Replacement Grammar based approach for AMR parsing. In *Proc. of CoNLL*, Beijing, China, pp. 32–41.

Ramisch, C., S. R. Cordeiro, A. Savary, V. Vincze, V. Barbu Mititelu, A. Bhatia, M. Buljan, M. Candito, P. Gantar, V. Giouli, T. Güngör, A. Hawwari, U. Iñurrieta, J. Kovalevskaitė, S. Krek, T. Lichte, C. Liebeskind, J. Monti, C. Parra Escartín, B. QasemiZadeh, R. Ramisch, N. Schneider, I. Stoyanova, A. Vaidya, and A. Walsh (2018, August). Edition 1.1 of the PARSEME Shared Task on Automatic

Identification of Verbal Multiword Expressions. In *Proc. of the Joint Workshop on Linguistic Annotation, Multiword Expressions and Constructions (LAW-MWE-CxG-2018)*, Santa Fe, New Mexico, USA, pp. 222–240.

Rozenberg, G. (1997). *Handbook of Graph Grammars and Comp.*, Volume 1. World scientific.

Steedman, M. (2000). *The Syntatic Process*. Cambridge, MA: MIT Press.

Szubert, I., A. Lopez, and N. Schneider (2018, June). A structured syntax-semantics interface for English-AMR alignment. In *Proc. of NAACL-HLT*, New Orleans, Louisiana, pp. 1169–1180.

Wang, C., B. Li, and N. Xue (2018, June). Transition-based Chinese AMR parsing. In *Proc. of NAACL-HLT*, New Orleans, Louisiana, pp. 247–252.

Wang, C., S. Pradhan, X. Pan, H. Ji, and N. Xue (2016, June). CAMR at SemEval-2016 Task 8: An extended transition-based AMR parser. In *Proc. of SemEval*, San Diego, California, USA, pp. 1173–1178.

Xue, N., O. Bojar, J. Hajič, M. Palmer, Z. Urešová, and X. Zhang (2014, May). Not an interlingua, but close: comparison of English AMRs to Chinese and Czech. In N. Calzolari, K. Choukri, T. Declerck, H. Loftsson, B. Maegaard, J. Mariani, A. Moreno, J. Odijk, and S. Piperidis (Eds.), *Proc. of LREC*, Reykjavík, Iceland, pp. 1765–1772.

A Additional Derivations

Below are full derivations illustrating raising, subject control, object control, an object control wh-question, a modal auxiliary with preposed VP adjunct, a purpose clause, coordinated purpose clauses, and right node raising with a shared main verb.

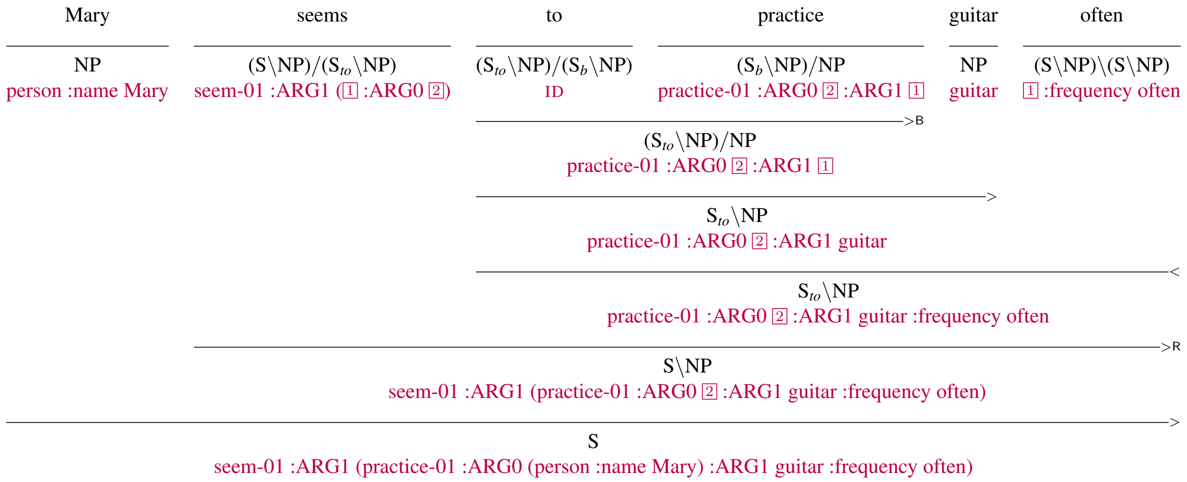

Mary
seems
to
practice
guitar
often
NP
person :name Mary
(S\NP)/(S_to\NP)
seem-01 :ARG1 ([1] :ARG0 [2])
(S_to\NP)/(S_b\NP)
ID
(S_b\NP)/NP
practice-01 :ARG0 [2] :ARG1 [1]
NP
guitar
(S\NP)\(S\NP)
[1] :frequency often
>B
(S_to\NP)/NP
practice-01 :ARG0 [2] :ARG1 [1]
>
S_to\NP
practice-01 :ARG0 [2] :ARG1 guitar
<
S_to\NP
practice-01 :ARG0 [2] :ARG1 guitar :frequency often
>R
S\NP
seem-01 :ARG1 (practice-01 :ARG0 [2] :ARG1 guitar :frequency often)
>
S
seem-01 :ARG1 (practice-01 :ARG0 (person :name Mary) :ARG1 guitar :frequency often)

Figure 9: Raising

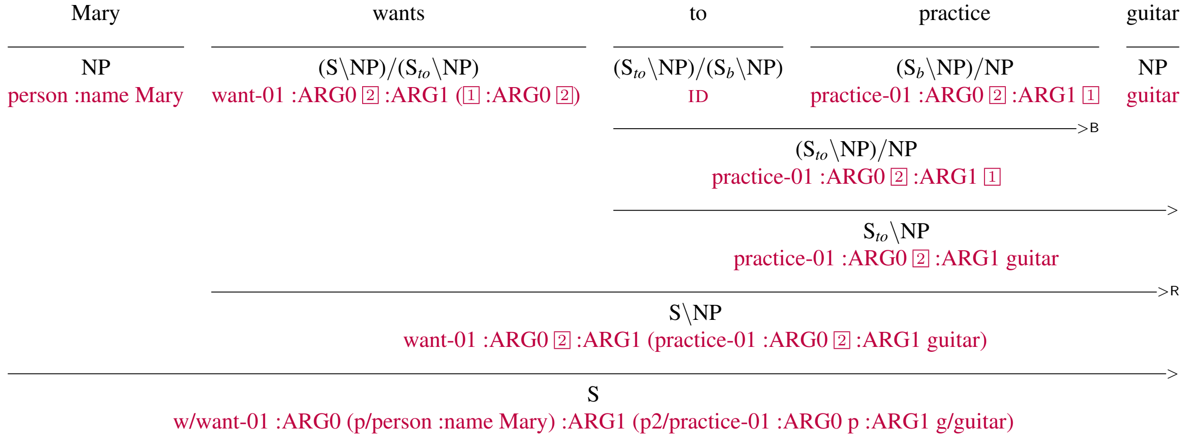

Mary
wants
to
practice
guitar
NP
person :name Mary
(S\NP)/(S_to\NP)
want-01 :ARG0 [2] :ARG1 ([1] :ARG0 [2])
(S_to\NP)/(S_b\NP)
ID
(S_b\NP)/NP
practice-01 :ARG0 [2] :ARG1 [1]
NP
guitar
>B
(S_to\NP)/NP
practice-01 :ARG0 [2] :ARG1 [1]
>
S_to\NP
practice-01 :ARG0 [2] :ARG1 guitar
>R
S\NP
want-01 :ARG0 [2] :ARG1 (practice-01 :ARG0 [2] :ARG1 guitar)
>
S
w/want-01 :ARG0 (p/person :name Mary) :ARG1 (p2/practice-01 :ARG0 p :ARG1 g/guitar)

Figure 10: Subject control

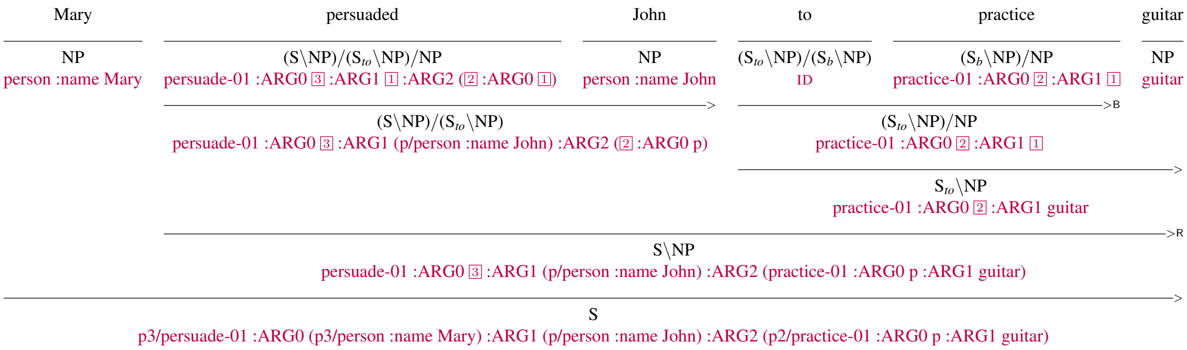

Mary
persuaded
John
to
practice
guitar
NP
person :name Mary
$(S\backslash NP)/(S_{to}\backslash NP)/NP$
persuade-01 :ARG0 $\boxed{3}$:ARG1 $\boxed{1}$:ARG2 ($\boxed{2}$:ARG0 $\boxed{1}$)
NP
person :name John
$(S_{to}\backslash NP)/(S_b\backslash NP)$
ID
$(S_b\backslash NP)/NP$
practice-01 :ARG0 $\boxed{2}$:ARG1 $\boxed{1}$
NP
guitar
>
$(S\backslash NP)/(S_{to}\backslash NP)$
persuade-01 :ARG0 $\boxed{3}$:ARG1 (p/person :name John) :ARG2 ($\boxed{2}$:ARG0 p)
>B
$(S_{to}\backslash NP)/NP$
practice-01 :ARG0 $\boxed{2}$:ARG1 $\boxed{1}$
>
$S_{to}\backslash NP$
practice-01 :ARG0 $\boxed{2}$:ARG1 guitar
>R
$S\backslash NP$
persuade-01 :ARG0 $\boxed{3}$:ARG1 (p/person :name John) :ARG2 (practice-01 :ARG0 p :ARG1 guitar)
>
S
p3/persuade-01 :ARG0 (p3/person :name Mary) :ARG1 (p/person :name John) :ARG2 (p2/practice-01 :ARG0 p :ARG1 guitar)

Figure 11: Object control. Note that the PropBank predicate persuade-01 specifies :ARG0 for the persuader, :ARG1 for the persuadee, and :ARG2 for the impelled action.

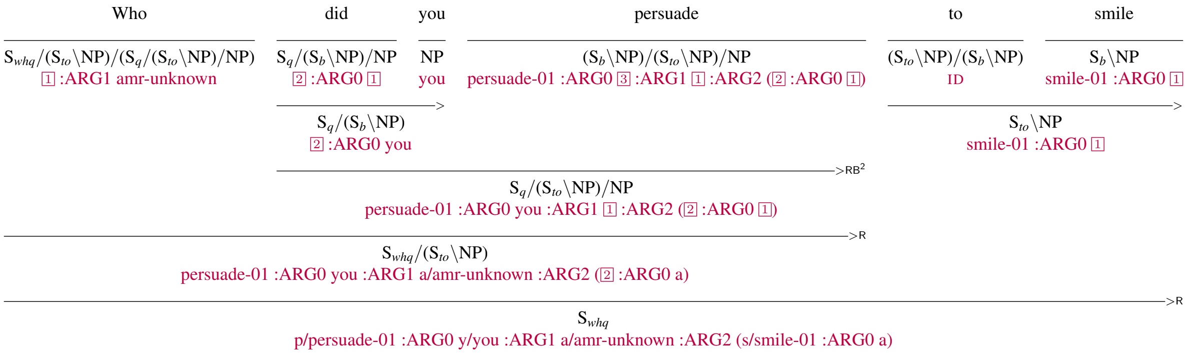

Who
did
you
persuade
to
smile
$S_{whq}/(S_{to}\backslash NP)/(S_q/(S_{to}\backslash NP)/NP)$
$\boxed{1}$:ARG1 amr-unknown
$S_q/(S_b\backslash NP)/NP$
$\boxed{2}$:ARG0 $\boxed{1}$
NP
you
$(S_b\backslash NP)/(S_{to}\backslash NP)/NP$
persuade-01 :ARG0 $\boxed{3}$:ARG1 $\boxed{1}$:ARG2 ($\boxed{2}$:ARG0 $\boxed{1}$)
$(S_{to}\backslash NP)/(S_b\backslash NP)$
ID
$S_b\backslash NP$
smile-01 :ARG0 $\boxed{1}$
>
$S_q/(S_b\backslash NP)$
$\boxed{2}$:ARG0 you
>
$S_{to}\backslash NP$
smile-01 :ARG0 $\boxed{1}$
>RB^2
$S_q/(S_{to}\backslash NP)/NP$
persuade-01 :ARG0 you :ARG1 $\boxed{1}$:ARG2 ($\boxed{2}$:ARG0 $\boxed{1}$)
>R
$S_{whq}/(S_{to}\backslash NP)$
persuade-01 :ARG0 you :ARG1 a/amr-unknown :ARG2 ($\boxed{2}$:ARG0 a)
>R
S_{whq}
p/persuade-01 :ARG0 y/you :ARG1 a/amr-unknown :ARG2 (s/smile-01 :ARG0 a)

Figure 12: Object control wh-question: "Who did you persuade to smile?" (example suggested by a reviewer)

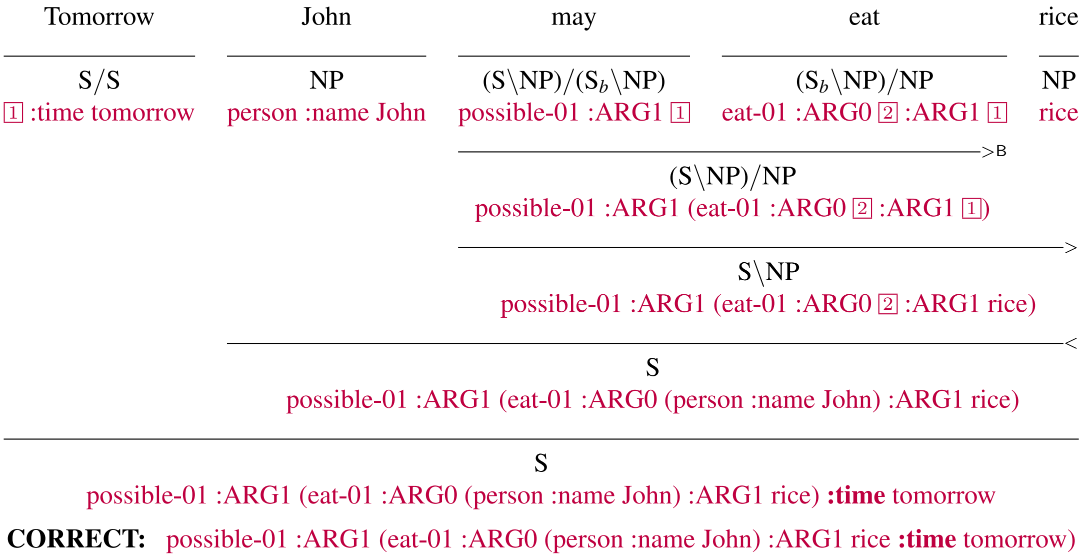

Tomorrow
John
may
eat
rice
S/S
$\boxed{1}$:time tomorrow
NP
person :name John
$(S\backslash NP)/(S_b\backslash NP)$
possible-01 :ARG1 $\boxed{1}$
$(S_b\backslash NP)/NP$
eat-01 :ARG0 $\boxed{2}$:ARG1 $\boxed{1}$
NP
rice
>B
$(S\backslash NP)/NP$
possible-01 :ARG1 (eat-01 :ARG0 $\boxed{2}$:ARG1 $\boxed{1}$)
>
$S\backslash NP$
possible-01 :ARG1 (eat-01 :ARG0 $\boxed{2}$:ARG1 rice)
<
S
possible-01 :ARG1 (eat-01 :ARG0 (person :name John) :ARG1 rice)
S
possible-01 :ARG1 (eat-01 :ARG0 (person :name John) :ARG1 rice) **:time** tomorrow
CORRECT: possible-01 :ARG1 (eat-01 :ARG0 (person :name John) :ARG1 rice **:time** tomorrow)

Figure 13: Modal auxiliary with preposed adjunct: "Tomorrow, John may eat rice". In the derived AMR, the temporal modifier is placed incorrectly under the modal predicate rather than the main event predicate.

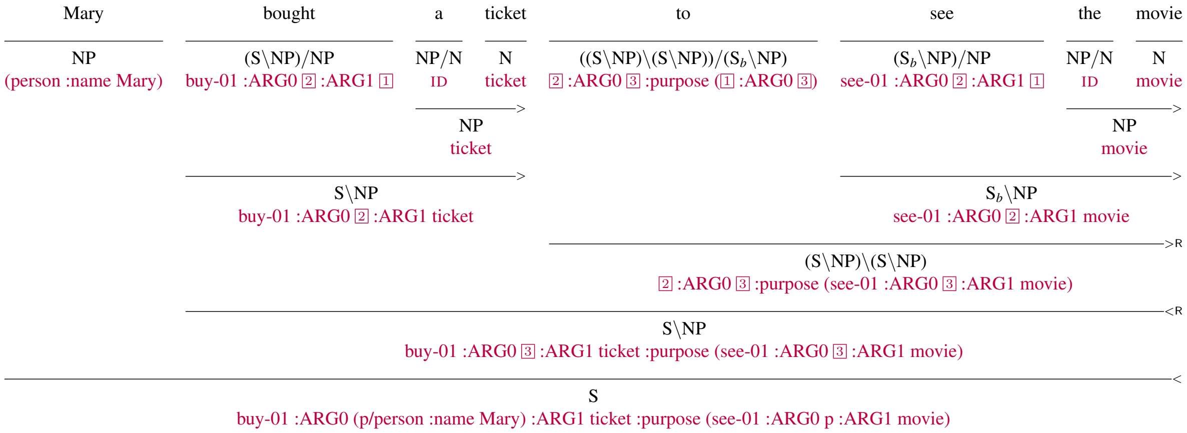

Mary bought a ticket to see the movie
NP (person :name Mary)
(S\NP)/NP buy-01 :ARG0 $\boxed{2}$:ARG1 $\boxed{1}$
NP/N ID
N ticket
((S\NP)\(S\NP))/(S_b\NP) $\boxed{2}$:ARG0 $\boxed{3}$:purpose ($\boxed{1}$:ARG0 $\boxed{3}$)
(S_b\NP)/NP see-01 :ARG0 $\boxed{2}$:ARG1 $\boxed{1}$
NP/N ID
N movie
>
NP ticket
>
NP movie
>
S\NP buy-01 :ARG0 $\boxed{2}$:ARG1 ticket
>
S_b\NP see-01 :ARG0 $\boxed{2}$:ARG1 movie
>R
(S\NP)\(S\NP) $\boxed{2}$:ARG0 $\boxed{3}$:purpose (see-01 :ARG0 $\boxed{3}$:ARG1 movie)
<R
S\NP buy-01 :ARG0 $\boxed{3}$:ARG1 ticket :purpose (see-01 :ARG0 $\boxed{3}$:ARG1 movie)
<
S buy-01 :ARG0 (p/person :name Mary) :ARG1 ticket :purpose (see-01 :ARG0 p :ARG1 movie)

Figure 14: *to*-purpose

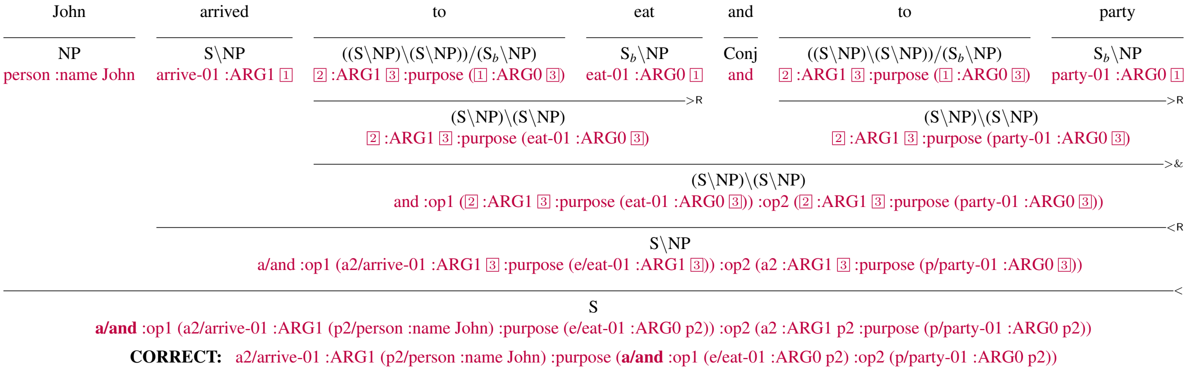

John arrived to eat and to party
NP person :name John
S\NP arrive-01 :ARG1 $\boxed{1}$
((S\NP)\(S\NP))/(S_b\NP) $\boxed{2}$:ARG1 $\boxed{3}$:purpose ($\boxed{1}$:ARG0 $\boxed{3}$)
S_b\NP eat-01 :ARG0 $\boxed{1}$
Conj and
((S\NP)\(S\NP))/(S_b\NP) $\boxed{2}$:ARG1 $\boxed{3}$:purpose ($\boxed{1}$:ARG0 $\boxed{3}$)
S_b\NP party-01 :ARG0 $\boxed{1}$
>R
(S\NP)\(S\NP) $\boxed{2}$:ARG1 $\boxed{3}$:purpose (eat-01 :ARG0 $\boxed{3}$)
>R
(S\NP)\(S\NP) $\boxed{2}$:ARG1 $\boxed{3}$:purpose (party-01 :ARG0 $\boxed{3}$)
>&
(S\NP)\(S\NP) and :op1 ($\boxed{2}$:ARG1 $\boxed{3}$:purpose (eat-01 :ARG0 $\boxed{3}$)) :op2 ($\boxed{2}$:ARG1 $\boxed{3}$:purpose (party-01 :ARG0 $\boxed{3}$))
<R
S\NP a/and :op1 (a2/arrive-01 :ARG1 $\boxed{3}$:purpose (e/eat-01 :ARG1 $\boxed{3}$)) :op2 (a2 :ARG1 $\boxed{3}$:purpose (p/party-01 :ARG0 $\boxed{3}$))
<
S **a/and** :op1 (a2/arrive-01 :ARG1 (p2/person :name John) :purpose (e/eat-01 :ARG0 p2)) :op2 (a2 :ARG1 p2 :purpose (p/party-01 :ARG0 p2))
CORRECT: a2/arrive-01 :ARG1 (p2/person :name John) :purpose (**a/and** :op1 (e/eat-01 :ARG0 p2) :op2 (p/party-01 :ARG0 p2))

Figure 15: Coordinated purpose clauses: “John arrived to eat and to party”. Note that the PropBank predicate arrive-01 has no :ARG0; its subject is :ARG1. The lexical semantics for infinitive purpose *to* is chosen accordingly. However, the placement in the derived AMR of the semantic conjunction and is incorrect.

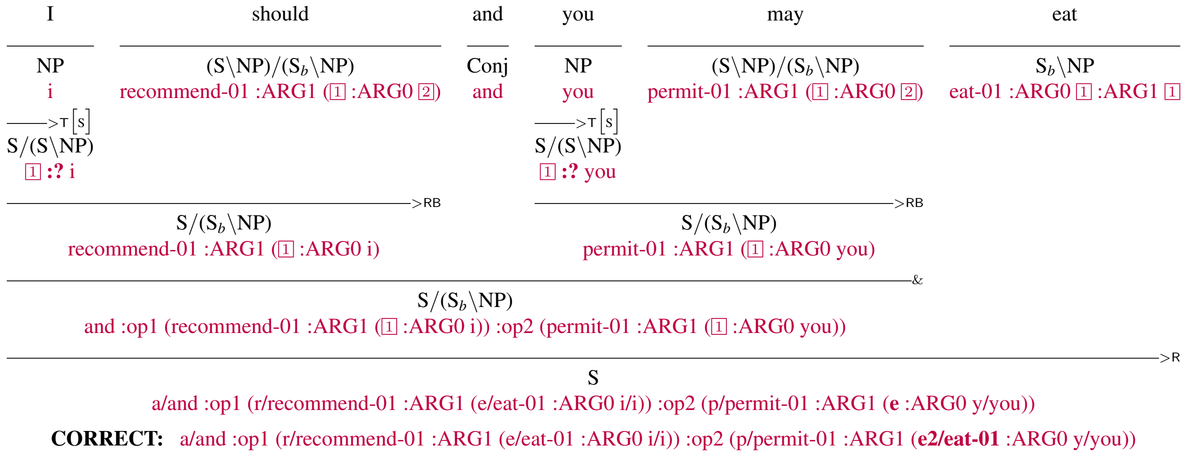

I should and you may eat
NP i
(S\NP)/(S_b\NP) recommend-01 :ARG1 ($\boxed{1}$:ARG0 $\boxed{2}$)
Conj and
NP you
(S\NP)/(S_b\NP) permit-01 :ARG1 ($\boxed{1}$:ARG0 $\boxed{2}$)
S_b\NP eat-01 :ARG0 $\boxed{1}$:ARG1 $\boxed{1}$
>T[s]
S/(S\NP) $\boxed{1}$ **:?** i
>T[s]
S/(S\NP) $\boxed{1}$ **:?** you
>RB
S/(S_b\NP) recommend-01 :ARG1 ($\boxed{1}$:ARG0 i)
>RB
S/(S_b\NP) permit-01 :ARG1 ($\boxed{1}$:ARG0 you)
&
S/(S_b\NP) and :op1 (recommend-01 :ARG1 ($\boxed{1}$:ARG0 i)) :op2 (permit-01 :ARG1 ($\boxed{1}$:ARG0 you))
>R
S a/and :op1 (r/recommend-01 :ARG1 (e/eat-01 :ARG0 i/i)) :op2 (p/permit-01 :ARG1 (**e** :ARG0 y/you))
CORRECT: a/and :op1 (r/recommend-01 :ARG1 (e/eat-01 :ARG0 i/i)) :op2 (p/permit-01 :ARG1 (**e2/eat-01** :ARG0 y/you))

Figure 16: Right node raising with shared main verb: “I should and you may eat”. The derived AMR has a reentrancy for the eat-01 predicate where there should be a separate copy of the predicate.